NUBE PRIVADA INTEGRAL: Virtualización, Identidad Centralizada y Servicios de Próxima Generación

Guía avanzada para el despliegue de infraestructuras seguras con KVM, LDAP y Authentik SSO

Arcadio Ortega Reinoso

KVM

NotebookLM

Virtualización, Identidad Centralizada y Servicios de Próxima Generación

Guía avanzada para el despliegue de infraestructuras seguras con KVM, LDAP y Authentik SSO.

Resumen

Implementación de una infraestructura de servidores avanzada y basada en un entorno de virtualización con KVM y QEMU.
La arquitectura centraliza la gestión de identidades mediante un directorio LDAP integrado con Authentik para ofrecer servicios de Single Sign-On (SSO).
El sistema despliega diversas máquinas virtuales especializadas, incluyendo plataformas de colaboración como Nextcloud y OpenProject, así como bases de datos en MariaDB y PostgreSQL.
Se incluyen guías detalladas para la configuración de redes bridge, gestión de almacenamiento LVM, seguridad mediante certificados SSL y automatización de servicios con scripts.
Finalmente, el texto aborda la monitorización del entorno a través de herramientas de observabilidad como Grafana y Loki.

Análisis del documento

El documento detalla la construcción de una nube privada escalable y segura, estructurada desde el hardware base hasta la entrega de servicios finales mediante una arquitectura de identidad dual.

1. El Hilo Conductor: De la Infraestructura al Servicio

El hilo conductor del documento es el **despliegue de una**

infraestructura de virtualización robusta diseñada para albergar servicios interconectados bajo un sistema de **identidad centralizada**. La narrativa progresa lógicamente: primero se prepara el "suelo" (el hipervisor y el almacenamiento), luego se instalan los "cimientos" (red, DNS y certificados), se levanta la "estructura de identidad" (LDAP y Authentik) y finalmente se despliegan las "habitaciones" o servicios (Nextcloud, bases de datos y herramientas de observabilidad).

2. Fase I: Preparación del Host de Virtualización (KVM/QEMU)

La infraestructura comienza con la transformación de un servidor físico (`Server01`) en un host de virtualización utilizando **KVM/QEMU**.

- **Gestión de Almacenamiento:** Se utiliza **LVM (Logical Volume Manager)** para gestionar el espacio de disco de forma flexible. Se crea un Volume Group (`vm-vg`) y un Logical Volume (`vm-storage`) de 900GB montado en `/vm`, lo que permite un manejo eficiente de las imágenes de las máquinas virtuales (VMs).
- **Pool de Almacenamiento y Permisos:** Se define un *storage pool* en libvirt para automatizar el acceso a las imágenes `.qcow2`. Un detalle técnico crítico es la modificación de los perfiles de **AppArmor** para permitir que QEMU acceda al directorio no estándar `/vm/libvirt/images`, garantizando la seguridad sin sacrificar la funcionalidad.
- **Red en Modo Bridge:** Para que las VMs sean visibles en la red física y no queden aisladas tras una NAT, se configura un **bridge (`br0`)** mediante Netplan. Esto permite que cada servicio tenga su propia IP en el rango de la red local (172.16.6.x), facilitando la resolución de nombres y el acceso externo.

3. Fase II: Automatización y Despliegue de Máquinas Virtuales

El documento destaca por el uso de **plantillas y automatización**

para acelerar el despliegue:

- **Plantillas (Templates):** Se crean imágenes base de Ubuntu 22.04 y 24.04 limpias, con `qemu-guest-agent` instalado para permitir la comunicación fluida con el hipervisor.
- **Scripting de Despliegue:** Se incluye un script de Bash (`instala_sistema.sh`) que actúa como un orquestador "mini-cloud". Este script utiliza `qemu-img` para crear clones rápidos (*thin provisioning*) y **cloud-init** para pre-configurar usuarios, claves SSH y redes en cada nueva VM al arrancar.

4. Fase III: Estrategia de Identidad (La Clave del Análisis)

El núcleo del proyecto es la separación de la gestión de identidades en dos capas diferenciadas:

A. La Capa Técnica: LDAP (Usuarios POSIX)

El sistema utiliza un directorio **LDAP** para gestionar los **usuarios internos o técnicos**.

- **Función:** Estos usuarios están destinados al acceso a sistemas y recursos internos de la red.
- **Implementación:** Se configuran objetos `posixAccount` y `posixGroup`. Esto permite, por ejemplo, que servicios como OpenProject se conecten directamente a LDAP para autenticar a trabajadores mediante atributos estándar como `uid`, `cn` y `mail`.
- **Seguridad:** Se implementa **LDAPS (LDAP sobre SSL)** utilizando una entidad certificadora (CA) propia, asegurando que las credenciales no viajen en texto plano por la red interna.

B. La Capa de Usuario Final: Authentik (Usuarios no POSIX)

Para los servicios orientados a la web y usuarios finales, se despliega **Authentik** sobre Docker en la VM de identidad.

- **Función:** Authentik actúa como un proveedor de identidad (IdP) moderno, ofreciendo una experiencia de **SSO (Single Sign-On)**.

- **Separación:** Esto permite distinguir claramente entre perfiles de trabajadores (sistemas) y clientes o usuarios de aplicaciones web.
- **Integración:** Authentik se alimenta de LDAP como fuente de datos, pero entrega la autenticación a través de protocolos modernos como **OIDC (OpenID Connect)**.

5. Fase IV: Despliegue de Servicios y Seguridad

Con la base de identidad lista, el documento procede a desplegar servicios integrados:

- **Infraestructura de Certificados:** Se crea una CA interna (`iespoveda.pem`) para firmar certificados SSL para todos los servicios (`auth`, `cloud`, `database`, `ldap`), garantizando comunicaciones cifradas en toda la infraestructura.
- **Nextcloud:** Se instala como plataforma de colaboración y se integra con Authentik mediante OIDC, permitiendo que los usuarios accedan sin tener que recordar múltiples contraseñas.
- **Bases de Datos Centralizadas:** Se dedica una VM específica para MariaDB y PostgreSQL, configurada para **exigir conexiones SSL** obligatorias para cualquier acceso remoto, reforzando la seguridad de los datos.
- **Observabilidad y Automatización:** Se introducen herramientas como **Grafana, Loki y Promtail** para la gestión de logs, y **n8n** para la automatización de flujos de trabajo (triggers de chat, CRM, etc.), integrando la monitorización de la infraestructura en un solo panel.

6. Fase V: Gobernanza Operativa

El documento cierra con la gestión del ciclo de vida de la infraestructura:

- **DNS/DHCP con Dnsmasq:** Se configura en el host para proporcionar resolución de nombres local y asignación de IPs fijas basadas en la MAC de las VMs.
- **Orden de Arranque y Parada:** Un script de gestión

(`gestion-vms.sh`) y un servicio de Systemd aseguran que las VMs arranquen en el orden correcto (primero base de datos e identidad, luego aplicaciones) y se apaguen de forma ordenada, evitando la corrupción de datos y fallos en las dependencias de los servicios.

Carácter Educativo y Autoría

Este documento es un cuaderno de apuntes y recursos editado con fines estrictamente educativos y de divulgación gratuita.
Una parte significativa de los ejercicios prácticos, explicaciones y desarrollos de código incluidos en esta obra son creaciones originales del autor.
El resto del contenido se compone de adaptaciones y fragmentos de código abierto recopilados de fuentes públicas de Internet.
El autor no reclama la propiedad intelectual de estos materiales de terceros y apela al derecho de cita con fines docentes.
Si usted es autor de alguno de los materiales externos incluidos y considera que su uso infringe sus derechos, por favor, póngase en contacto con el autor para su inmediata modificación o retirada en la siguiente revisión del archivo.

Licencia de Uso

Este libro de apuntes se distribuye de forma totalmente gratuita bajo la licencia:
Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional (CC BY-NC-SA 4.0).
Usted es libre de:

- Compartir: Copiar y redistribuir el material en cualquier soporte o formato.
- Adaptar: Remezclar, transformar y construir a partir del material.

Bajo los siguientes términos:

- Atribución: Debe reconocer la autoría de forma adecuada, proporcionar un enlace a la licencia e indicar si se han realizado cambios.
- No Comercial: No puede utilizar el material con fines comerciales ni de venta directa.
- Compartir Igual: Si remezcla, transforma o crea a partir de este material, debe distribuir su contribución bajo la misma licencia que el original.

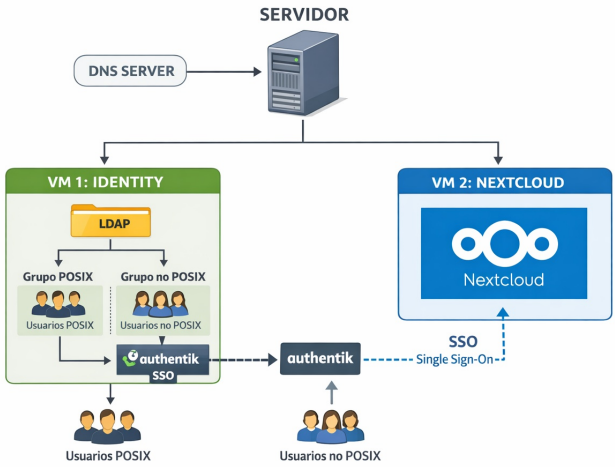
La arquitectura se compone de un servidor principal que actúa como host de virtualización y proporciona resolución de nombres a través de un servicio DNS. Sobre este servidor se despliegan dos máquinas virtuales diferenciadas según su función: (Entre otros)

- Identity: encargada de la gestión de identidades, donde se implementa un directorio LDAP junto con Authentik como proveedor de identidad (IdP).
- Nextcloud: plataforma de almacenamiento y colaboración accesible para los usuarios finales.

El sistema de directorio LDAP organiza a los usuarios en dos categorías:

- Usuarios POSIX: destinados a acceso a sistemas y recursos internos de la red.
- Usuarios no POSIX: orientados al acceso a servicios web.

Estos últimos se integran con Authentik, que actúa como proveedor de autenticación centralizada, ofreciendo acceso mediante SSO a la plataforma Nextcloud.
De este modo, se consigue separar claramente los perfiles de usuarios internos (trabajadores) de los externos (clientes), mejorando tanto la seguridad como la gestión de accesos.



Logramos:

- Separación de servicios en máquinas virtuales (mejor aislamiento y escalabilidad).
- Uso de LDAP como fuente central de identidad (clave en entornos medianos/grandes).
- Distinción entre usuarios POSIX y no POSIX (muy útil para separar acceso a sistemas vs. servicios).
- Implementación de SSO con Authentik (mejora mucho la experiencia de usuario y la seguridad).

```
sudo apt update
sudo apt upgrade
sudo apt install -y qemu-kvm libvirt-daemon-system
libvirt-clients bridge-utils virtinst cloud-image-utils
docker.io docker-compose-v2
```

```
admon@srv01:~$ sudo adduser $USER libvirt
info: The user `admon' is already a member of `libvirt'.
```

`admon@srv01:~$ sudo adduser $USER kvm` info: Adding user

`admon' to group `kvm' ...

```
admon@srv01:~$ sudo adduser $USER docker
info: Adding user `admon' to group `docker' ...
```

```
admon@srv01:~$ df -hT
Filesystem                         Type   Size  Used
Avail Use% Mounted on
tmpfs                              tmpfs  3,1G  2,3M
3,1G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv ext4    98G   12G
82G  13% /
tmpfs                              tmpfs   16G     0
16G   0% /dev/shm
tmpfs                              tmpfs  5,0M     0
5,0M   0% /run/lock
/dev/sdb2                          ext4   2,0G  104M
1,7G   6% /boot
tmpfs                              tmpfs  3,1G   12K
3,1G   1% /run/user/1000
tmpfs                              tmpfs   16G     0
16G   0% /run/qemu
```

```
admon@srv01:~$ sudo lsblk
NAME                      MAJ:MIN RM   SIZE RO TYPE
MOUNTPOINTS
sda                         8:0    0 931,5G  0 disk
sdb                         8:16   0   1,1T  0 disk
  sdb1                      8:17   0     1M  0 part
  sdb2                      8:18   0     2G  0 part /
boot
  sdb3                      8:19   0   1,1T  0 part
    ubuntu--vg-ubuntu--lv 252:0    0   100G  0 lvm  /
```

```
admon@srv01:~$ sudo pvcreate /dev/sda
  Physical volume "/dev/sda" successfully created.

admon@srv01:~$ sudo vgcreate vm-vg /dev/sda
  Volume group "vm-vg" successfully created

admon@srv01:~$ sudo lvcreate -L 900G -n vm-storage vm-vg
  Logical volume "vm-storage" created.
```

Ahora tenemos:

```
admon@srv01:~$ sudo lsblk
NAME                      MAJ:MIN RM   SIZE RO TYPE
MOUNTPOINTS
sda                         8:0    0 931,5G  0 disk
  vm--vg-vm--storage      252:1    0   900G  0 lvm
sdb                         8:16   0   1,1T  0 disk
  sdb1                      8:17   0     1M  0 part
  sdb2                      8:18   0     2G  0 part /
boot
  sdb3                      8:19   0   1,1T  0 part
    ubuntu--vg-ubuntu--lv 252:0    0   100G  0 lvm  /
```

Luego tengo:

- Disco sda
- convertido a Physical Volume (PV)
- dentro del Volume Group vm-vg
- con un Logical Volume vm-storage de 900GB

Ahora creamos file system y lo montamos:
---------- Todo cambiado al siguiente disco

```
sudo lvcreate -L 500G -n vmraid ubuntu-vg
sudo mkfs.ext4 /dev/ubuntu-vg/vmraid
sudo mkdir /vmraid
sudo mount /dev/ubuntu-vg/vmraid /vmraid
echo "Hay que cambiar entrada en fstab. Después de lo
siguiente"
sudo umount /vm
sudo mount /dev/ubuntu-vg/vmraid /vm
```

```
admon@srv01:~$ sudo mkfs.ext4 /dev/vm-vg/vm-storage
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 235929600 4k blocks and
58982400 inodes
Filesystem UUID: ded47898-f6ce-451b-82d8-0dccc8577bc2
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200,
884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872,
71663616, 78675968,
```

```
        102400000, 214990848

Allocating group tables: done

Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting
information: done
```

```
admon@srv01:~$ sudo mkdir /vm
admon@srv01:~$ sudo mount /dev/vm-vg/vm-storage /vm
admon@srv01:~$ df -h
Filesystem                         Size  Used Avail Use%
Mounted on
tmpfs                              3,1G  2,3M  3,1G   1%
/run
/dev/mapper/ubuntu--vg-ubuntu--lv   98G   12G   82G  13%
/
tmpfs                               16G     0   16G   0%
/dev/shm
tmpfs                              5,0M     0  5,0M   0%
/run/lock
/dev/sdb2                          2,0G  104M  1,7G   6%
/boot
tmpfs                              3,1G   12K  3,1G   1%
/run/user/1000
tmpfs                               16G     0   16G   0%
/run/qemu
/dev/mapper/vm--vg-vm--storage     885G   28K  840G   1%
/vm
```

Lo añadimos al fstab:

```
admon@srv01:~$ sudo vi /etc/fstab
admon@srv01:~$ sudo cat /etc/fstab
# /etc/fstab: static file system information.
#
...
...
/dev/disk/by-uuid/df5db9d6-3095-43d9-99b7-8ba03ae7b2f6 /
boot ext4 defaults 0 1
/swap.img       none    swap    sw      0       0
#AOR# Para las maquinas virtuales
/dev/vm-vg/vm-storage   /vm   ext4   defaults   0  2
```

Ahora lo montamos todo

```
sudo mount -a
```

Resultado final:

```
admon@srv01:~$ df -hT
Filesystem                         Type   Size  Used
Avail Use% Mounted on
tmpfs                              tmpfs  3,1G  2,3M
3,1G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv ext4    98G   12G
82G  13% /
tmpfs                              tmpfs   16G     0
16G   0% /dev/shm
tmpfs                              tmpfs  5,0M     0
5,0M   0% /run/lock
/dev/sdb2                          ext4   2,0G  104M
1,7G   6% /boot
tmpfs                              tmpfs  3,1G   12K
3,1G   1% /run/user/1000
tmpfs                              tmpfs   16G     0
16G   0% /run/qemu
/dev/mapper/vm--vg-vm--storage     ext4   885G   28K
840G   1% /vm
```

Ahora creamos la infraestructura para nuestras máquinas virtuales:

Configurar KVM

Crear carpeta para las VMs

Libvirt normalmente usa: `/var/lib/libvirt/images` Pero nosotros usaremos:

```
sudo mkdir -p /vm/libvirt/images
```

Permisos:

```
```

```
sudo chown root:libvirt /vm/libvirt/images
sudo chmod 0775 /vm/libvirt/images
```

Crear el storage pool de KVM

Primero creamos el pool:

```
sudo virsh pool-define-as vm-pool dir - - - - "/vm/
libvirt/images"
```

Esto es:

```
admon@srv01:~$ sudo virsh pool-define-as vm-pool dir - -
- - "/vm/libvirt/images"
Pool vm-pool defined
```

Activarlo

```
sudo virsh pool-start vm-pool
Pool vm-pool started
```

Activarlo al arrancar

```
sudo virsh pool-autostart vm-pool
Pool vm-pool marked as autostarted
```

Con autostart:

```
arranque servidor
      ↓
libvirt arranca
      ↓
vm-pool se activa
      ↓
```

```
VMs pueden usar /vm/libvirt/images
```

Verificar

```
admon@srv01:~$ sudo virsh pool-list --all
 Name       State    Autostart
--------------------------------
 vm-pool    active   yes
```

Arquitectura final

```
sdb (1.1TB)
   /boot
   LVM
       ubuntu-lv (100G) → /

sda (1TB)
   LVM
       vm-storage (900G)
             /vm/libvirt/images
                  vm1.qcow2
                  vm2.qcow2
                  vm3.qcow2
```

Red bridge

Configuramos el bridge para que las VMs **NO** estén en una red NAT interna tipo: 192.168.122.x. Con bridge

```
internet/router

   switch

   br0 (bridge)

 host    VM1
         VM2
         VM3
```

Red actual

```
Interfaz: enp26s0
IP: 172.16.6.5
Máscara: /24
Red: 172.16.6.0/24
```

Configuramos netplan de nuevo

```
admon@srv01:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2

  ethernets:
    enp26s0:
      dhcp4: false

  bridges:
    br0:
      interfaces: [enp26s0]
      addresses:
        - 172.16.6.5/24
      nameservers:
        addresses:
          - 172.16.6.1
      routes:
        - to: default
          via: 172.16.6.1
      parameters:
        stp: false
        forward-delay: 0
```

```
admon@srv01:~$ sudo netplan apply
admon@srv01:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd
00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
...
...
```

```
8: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500
qdisc noqueue state DOWN group default
    link/ether 66:f9:32:82:a7:94 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global
docker0
       valid_lft forever preferred_lft forever
9: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500
qdisc noqueue state DOWN group default qlen 1000
    link/ether 52:54:00:e9:d4:21 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope
global virbr0
       valid_lft forever preferred_lft forever
10: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
qdisc noqueue state UP group default qlen 1000
    link/ether ba:de:75:55:49:ee brd ff:ff:ff:ff:ff:ff
    inet 172.16.6.5/24 brd 172.16.6.255 scope global br0
       valid_lft forever preferred_lft forever
    inet6 fe80::b8de:75ff:fe55:49ee/64 scope link
       valid_lft forever preferred_lft forever
```

```
admon@srv01:~$ ip route
default via 172.16.6.1 dev br0 proto static
172.16.6.0/24 dev br0 proto kernel scope link src
172.16.6.5
172.17.0.0/16 dev docker0 proto kernel scope link src
172.17.0.1 linkdown
192.168.122.0/24 dev virbr0 proto kernel scope link src
192.168.122.1 linkdown
```

Verificamos las conexiones:

```
admon@srv01:~$ ping -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=14.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=15.5 ms
```

Utilidades varias

Raids BIOS

```
```

```
sudo apt-get install dmraid
```

Importante permisos

`/etc/apparmor.d/abstractions/libvirt-qemu`**Al final del archivo (antes de include if exists <local/abstractions/libvirt-qemu>), añadimos: /vm/libvirt/images/** rwk,** El fichero tiene que quedar como sigue:

```
# required for QEMU accessing UEFI nvram variables
 owner /var/lib/libvirt/qemu/nvram/*_VARS.fd rwk,
 owner /var/lib/libvirt/qemu/nvram/*_VARS.ms.fd rwk,

 #AOR##
 /vm/libvirt/images/** rwk,

 include if exists <abstractions/libvirt-qemu.d>

 ### DEPRECATED ###
 # Debian-specific override file
```

Explicación:

- rwk → read, write, lock
- ** → todo lo que esté dentro del directorio, incluyendo subdirectorios

Esto permite que QEMU acceda a todos los discos en /vm/libvirt/images.

- Guardamos y cerramos el archivo.
- Recargar AppArmor. Para que los cambios tengan efecto:

```
sudo systemctl reload apparmor
```

También se puede hacer, como alternativa:

```
sudo apparmor_parser -r /etc/apparmor.d/
abstractions/libvirt-qemu
```

Esto recarga el perfil sin reiniciar el sistema.

- Verificamos los permisos de los discos.

```
sudo chown libvirt-qemu:kvm /vm/libvirt/images/
ubuntu2204-base.qcow2
sudo chmod 660 /vm/libvirt/images/ubuntu2204-
base.qcow2
```

Creamos MV y templates

Crear la primera VM base (Ubuntu 22.04)

Descarga la ISO de Ubuntu Server 22.04:

```
wget https://releases.ubuntu.com/22.04/ubuntu-22.04.5-
live-server-amd64.iso -P /vm/isos
```

Creamos un disco base para la VM (qcow2, thin provisioning), por ejemplo de 20G:

```
sudo qemu-img create -f qcow2 /vm/libvirt/images/
ubuntu2204-base.qcow2 20G
```

Desde donde vamos a realizar la instalación instalamos previamente: `sudo apt install virt-viewer` y ahora instalamos la VM usando la ISO:

```
sudo virt-install \
 --name ubuntu2204-base \
 --memory 2048 \
```

```
 --vcpus 2 \
 --disk path=/vm/libvirt/images/ubuntu2204-
base.qcow2,size=20 \
 --os-variant ubuntu22.04 \
 --cdrom /vm/isos/ubuntu-22.04.5-live-server-amd64.iso \
 --network bridge=br0 \
 --graphics vnc
WARNING  Gráficos solicitados, pero DISPLAY (pantalla)
no está configurado. No se está ejecutando virt-viewer.
WARNING  No hay consola que lanzar para el invitado
(guest), volviendo al valor predeterminado --wait -1

Empezando la instalación...
Creando dominio...
              |    0 B  00:00:00

Domain is still running. Installation may be in
progress.
```

Abrimos otra sesión en el nuevo servidor y hacemos:

```
admon@srv01:~$ virsh vncdisplay ubuntu2204-base
127.0.0.1:0
```

Entonces nos conectamos

```
aor@smr218:~$ virt-viewer --connect qemu+ssh://
admon@172.16.6.5/system ubuntu2204-base
```

y se abrirá la ventana de trabajo.
Instalamos Ubuntu normalmente. **Recomendación para plantilla:**
Usuario: `admon`
Una vez acabada la instalación en la pantalla de lanzamiento tenemos:

```
...
...
Domain is still running. Installation may be in
progress.
Waiting for the installation to complete.

El dominio ha concluido. Continuando.
Creación de dominio completada.
```

```
Reiniciando invitado.
```

Comandos útiles

```
admon@srv01:~$ virsh list --all
 Id   Name              State
--------------------------------
 3    ubuntu2204-base   running
```

Podemos hacer por ejemplo:

```
virsh shutdown ubuntu2204-base
virsh destroy ubuntu2204-base
```

En el caso de `destroy` posteriormente hacer también: `virsh undefine ubuntu2204-base` Y para ver la ip que tiene en estos momentos, por ejemplo

```
virsh domifaddr ubuntu2204-base
```

Ahora nos conectamos al server y lo actualizamos

qemu-guest-agent

Es un servicio (demonio) instalado dentro de una máquina virtual (invitado) que permite la comunicación directa con el hipervisor (como Proxmox VE o KVM). Facilita tareas de gestión cruciales, incluyendo el apagado/reinicio ordenado, la congelación del sistema de archivos para copias de seguridad consistentes y la obtención de la dirección IP

```
sudo apt install qemu-guest-agent -y
```

Limpiamos los logs y temporales para plantilla:

```
sudo cloud-init clean
sudo rm -rf /var/log/*
```

Convertir VM en plantilla

Apaga la VM:

```
sudo virsh shutdown ubuntu2204-base
```

Verificamos que está parado:

```
admon@srv01:~$ virsh list --all
 Id   Name              State
----------------------------------
 -    ubuntu2204-base   shut off
```

Crea **snapshot como plantilla**:

```
admon@srv01:~$ sudo virsh snapshot-create-as --domain
ubuntu2204-base ubuntu2204-template "Base template
Ubuntu 22.04" --disk-only --atomic
Domain snapshot ubuntu2204-template created
```

> *O podemos simplemente **duplicar el disco qcow2** para clonar:*
>
> ```
> cp /vm/libvirt/images/ubuntu2204-base.qcow2 /vm/
> libvirt/images/ubuntu2204-clone1.qcow2
> ```

Quedando de esta manera:

```
admon@srv01:~$ ls -lah /vm/libvirt/images/
total 7,3G
drwxrwxr-x 2 root libvirt 4,0K mar 13 10:01 .
drwxr-xr-x 3 root root    4,0K mar 11 11:54 ..
-rw-r--r-- 1 root root    7,3G mar 13 09:59 ubuntu2204-
base.qcow2
-rw------- 1 root root    193K mar 13 10:01 ubuntu2204-
base.ubuntu2204-template
```

Crear segunda plantilla (Ubuntu 24.04)

Repetimos el mismo proceso para ubuntu 24.04:

Descarga ISO:

```
wget https://releases.ubuntu.com/24.04/ubuntu-24.04.4-
live-server-amd64.iso -P /vm/isos
```

Creamos disco:

```
sudo qemu-img create -f qcow2 /vm/libvirt/images/
ubuntu2404-base.qcow2 20G
```

Instalamos el servidor:

```
sudo virt-install \
 --name ubuntu2404-base \
 --memory 2048 \
 --vcpus 2 \
 --disk path=/vm/libvirt/images/ubuntu2404-
base.qcow2,size=20 \
 --os-variant ubuntu24.04 \
 --cdrom /vm/isos/ubuntu-24.04.4-live-server-amd64.iso \
 --network bridge=br0 \
 --graphics vnc
WARNING  Requested memory 2048 MiB is less than the
recommended 3072 MiB for OS ubuntu24.04
WARNING  Gráficos solicitados, pero DISPLAY (pantalla)
no está configurado. No se está ejecutando virt-viewer.
WARNING  No hay consola que lanzar para el invitado
(guest), volviendo al valor predeterminado --wait -1

Empezando la instalación...
Allocating 'ubuntu2404-base.qcow2'
                          |    0 B  00:00:00 ...
Creando dominio...
                          |    0 B  00:00:00

Domain is still running. Installation may be in
progress.
Waiting for the installation to complete.
```

Abrimos otra sesión en el servidor y hacemos:

```
admon@srv01:~$ virsh vncdisplay ubuntu2404-base
127.0.0.1:0
```

Entonces nos conectamos a:

```
aor@smr218:~$ virt-viewer --connect qemu+ssh://
admon@172.16.6.5/system ubuntu2404-base
```

y se abrirá la ventana de trabajo.
Instalamos Ubuntu normalmente. **Recomendación para plantilla:**
Usuario: `admon`
Una vez acabada la instalación en la pantalla de lanzamiento tenemos:

```
...
...
El dominio ha concluido. Continuando.
Creación de dominio completada.
Reiniciando invitado.
```

Ahora actualizamos, instalamos `qemu-guest-agent` y limpiamos logs.

```
admon@srv01:~$ virsh list --all
 Id   Name              State
--------------------------------
 2    ubuntu2404-base   running
 -    ubuntu2204-base   shut off
```

Limpiamos los logs y temporales para plantilla:

```
sudo apt update
sudo apt upgrade
sudo apt install qemu-guest-agent -y
sudo cloud-init clean
sudo rm -rf /var/log/*
sudo shutdown -h now
```

Apaga y duplica disco para clonar más adelante.

```
admon@srv01:~$ sudo virsh snapshot-create-as --domain
ubuntu2404-base ubuntu2404-template "Base template
Ubuntu 24.04" --disk-only --atomic
Domain snapshot ubuntu2404-template created
```

```
admon@srv01:~$ ls -lah /vm/libvirt/images/
total 12G
drwxrwxr-x 2 root libvirt 4,0K mar 13 10:47 .
drwxr-xr-x 3 root root    4,0K mar 11 11:54 ..
-rw-r--r-- 1 root root    7,3G mar 13 09:59 ubuntu2204-
base.qcow2
-rw------- 1 root root    193K mar 13 10:01 ubuntu2204-
base.ubuntu2204-template
-rw------- 1 root root     21G mar 13 10:40 ubuntu2404-
base.qcow2
-rw------- 1 root root    193K mar 13 10:47 ubuntu2404-
base.ubuntu2404-template
```

Clonar VMs rápidamente

Para clonar usando **qcow2 (thin)**:

```
sudo qemu-img create -f qcow2 -b /vm/libvirt/images/
ubuntu2404-base.qcow2 /vm/libvirt/images/ubuntu2404-
clone1.qcow2
```

`-b` → referencia la plantilla base (ahorra espacio, thin provisioning).

Ejemplo:

```
sudo virt-install \
--name ubuntu2404-clone1 \
--memory 2048 \
--vcpus 2 \
--disk path=/vm/libvirt/images/ubuntu2404-
clone1.qcow2,format=qcow2 \
--os-variant ubuntu24.04 \
--network bridge=br0,model=virtio \
```

```
--graphics none \
--console pty,target_type=serial
```

Utilidades

Encender:

```
sudo virsh start ubuntu2204-base
```

```
admon@srv01:~$ sudo virsh list --all
 Id   Name              State
--------------------------------
 1    ubuntu2204-base   running
 2    ubuntu2404-base   running
```

Apagar limpio (gracias al guest agent):

```
virsh shutdown ubuntu2204-base
```

Apagado forzado (como cortar corriente):

```
virsh destroy ubuntu2204-base
```

Reiniciar:

```
sudo virsh reboot ubuntu2204-base
Domain 'ubuntu2204-base' is being rebooted
```

Ver la IP de la VM (requiere qemu-guest-agent). Para ver direcciones ethern e ip

```
admon@srv01:~$ sudo virsh domiflist ubuntu2204-base
 Interface   Type     Source   Model    MAC
-------------------------------------------------------
 vnet0       bridge   br0      virtio
```

```
52:54:00:63:9f:9a

admon@srv01:~$ sudo virsh domifaddr ubuntu2204-base --
source agent
 Name       MAC address          Protocol     Address
-------------------------------------------------------------------------------
 lo         00:00:00:00:00:00    ipv4
127.0.0.1/8
 -          -                    ipv6         ::1/128
 enp1s0     52:54:00:63:9f:9a    ipv4
172.16.6.191/24
 -          -                    ipv6
fe80::5054:ff:fe63:9f9a/64
```

Si queremos entrar no vía ssh sino a nivel de consola

```
sudo virsh console ubuntu2204-base
```

Salir:

```
Ctrl + ]
```

Para poder entrar primero tenemos que hacer:

- Activar login en la consola serie

  ```
  sudo systemctl enable serial-getty@ttyS0.service
  sudo systemctl start serial-getty@ttyS0.service
  ```

  Comprueba:

  ```
  systemctl status serial-getty@ttyS0
  ```

  Debería aparecer active (running).
- Asegurar que el kernel usa la consola serie
  - Edita GRUB:

    ```
    sudo nano /etc/default/grub
    ```

- Busca:

```
GRUB_CMDLINE_LINUX=""
```

- y cámbialo a:

```
GRUB_CMDLINE_LINUX="console=tty0
console=ttyS0,115200n8"
```

- Actualizar grub

```
sudo update-grub
```

- Reiniciar la VM

```
sudo reboot
```

- Probar otra vez desde el host

```
sudo virsh console ubuntu2204-base
```

Ahora debería aparecer:

- Ubuntu 22.04 LTS ubuntu2204-base ttyS0
- ubuntu2204-base login

Ver información de la VM

```
admon@srv01:~$ sudo virsh dominfo ubuntu2204-base
Id:             1
Name:           ubuntu2204-base
UUID:           457...6693c
OS Type:        hvm
State:          running
CPU(s):         2
CPU time:       54,7s
Max memory:     2097152 KiB
Used memory:    2097152 KiB
Persistent:     yes
Autostart:      disable
```

```
Managed save:   no
Security model: apparmor
Security DOI:   0
Security label: libvirt-457...6693c (enforcing)
```

```
sudo virsh dumpxml ubuntu2204-base
<domain type='kvm' id='1'>
  <name>ubuntu2204-base</name>
  <uuid>457...6693c</uuid>
  <metadata>
...
...
    <imagelabel>+64055:+994</imagelabel>
  </seclabel>
</domain>
```

Snapshots

```
virsh snapshot-create-as ubuntu2204-base snap1
```

```
admon@srv01:~$ sudo virsh snapshot-list ubuntu2204-base
 Name                 Creation Time               State
------------------------------------------------------------
 ubuntu2204-template   2026-03-13 10:01:39 +0000
shutoff
```

Restaurar snapshot:

```
virsh snapshot-revert ubuntu2204-base snap1
```

Backups

Backups consistentes usando guest agent.
El guest agent permite congelar el filesystem antes del backup.

- Congelar (Esto evita corrupción en bases de datos o sistemas

activos):

```
virsh domfsfreeze ubuntu2204-base
```

- Hacer copia del disco (ejemplo):

```
cp /vm/libvirt/images/ubuntu2204-base.qcow2 /
backup/
```

- Descongelar:

```
virsh domfsthaw ubuntu2204-base
```

MVs. Ldap. Authentik. Nextcloud

Máquinas virtuales

Una vez que tenemos el servidor con los templates generados comenzamos a crear nuestras máquinas virtuales

```
#!/bin/bash
set -e

LOG="/var/log/mini-cloud.log"

BRIDGE="br0"
STORAGE="/vm/libvirt/images"

TEMPLATE2204="$STORAGE/ubuntu2204-base.ubuntu2204-
template"
TEMPLATE2404="$STORAGE/ubuntu2404-base.ubuntu2404-
template"

DOMAIN="informatica.iespoveda.com"
GATEWAY="172.16.6.1"
DNS="172.16.6.5"

SSH_HOME="/home/admon"

# Detectar usuario original y su clave pública
if [ -n "$SUDO_USER" ]; then
    USER_HOME=$(eval echo "~$SUDO_USER")
else
    USER_HOME="$HOME"
fi

if [ ! -f "$USER_HOME/.ssh/id_rsa.pub" ]; then
    echo "ERROR: No se encontró la clave pública en
$USER_HOME/.ssh/id_rsa.pub"
    echo "Genera una con: ssh-keygen -t rsa -b 4096"
    exit 1
```

```
fi

SSH_KEY=$(cat "$SSH_HOME/.ssh/id_rsa.pub")
#log "Usando clave pública de $USER_HOME/.ssh/
id_rsa.pub"

VM_NAMES=(
identity
database
core
nextcloud
openproject
fediverse
observability
)

declare -A VM_IP
declare -A VM_RAM
declare -A VM_CPU
declare -A VM_OS

VM_IP[identity]="172.16.6.10"
VM_IP[database]="172.16.6.20"
VM_IP[core]="172.16.6.30"
VM_IP[nextcloud]="172.16.6.40"
VM_IP[openproject]="172.16.6.41"
VM_IP[fediverse]="172.16.6.50"
VM_IP[observability]="172.16.6.60"

VM_RAM[identity]=3072
VM_RAM[database]=4096
VM_RAM[core]=3072
VM_RAM[nextcloud]=4096
VM_RAM[openproject]=4096
VM_RAM[fediverse]=8192
VM_RAM[observability]=4096

VM_CPU[identity]=2
VM_CPU[database]=2
VM_CPU[core]=2
VM_CPU[nextcloud]=2
VM_CPU[openproject]=2
VM_CPU[fediverse]=4
VM_CPU[observability]=2

VM_OS[identity]="24"
VM_OS[database]="24"
```

```
VM_OS[core]="24"
VM_OS[nextcloud]="24"
VM_OS[openproject]="22"
VM_OS[fediverse]="24"
VM_OS[observability]="24"

log(){
echo "[INFO] $1"
echo "[INFO] $1" >> $LOG
}

pause(){
read -p "Press ENTER"
}

check_root(){
if [ "$EUID" -ne 0 ]; then
echo "Run as root"
exit
fi
}

detect_host(){
log "Detecting host hardware"
echo "CPU:"
nproc
echo "RAM:"
free -h
echo "Disk:"
df -h
}



install_host_packages(){

log "Installing host packages"
apt update
apt install -y \
qemu-kvm \
libvirt-daemon-system \
virtinst \
cloud-image-utils \
qemu-utils \
docker.io \
docker-compose \
dnsmasq \
```

```
curl \
wget \
git
systemctl enable libvirtd
systemctl start libvirtd
}

clone_disk(){
VM=$1
if [ "${VM_OS[$VM]}" == "22" ]
then
TEMPLATE=$TEMPLATE2204
else
TEMPLATE=$TEMPLATE2404
fi
log "Creating disk for $VM"
qemu-img create \
-f qcow2 \
-F qcow2 \
-b $TEMPLATE \
$STORAGE/$VM.qcow2
}

create_cloud_init(){
VM=$1
IP=${VM_IP[$VM]}
mkdir -p /tmp/$VM

cat > /tmp/$VM/user-data <<EOF
#cloud-config

hostname: $VM

users:
  - name: ubuntu
    sudo: ALL=(ALL) NOPASSWD:ALL
    groups: sudo
    shell: /bin/bash
    ssh-authorized-keys:
      - $SSH_KEY

packages:
  - docker.io
  - docker-compose
  - qemu-guest-agent

runcmd:
```

```
  - systemctl enable docker
  - systemctl start docker
  - systemctl enable qemu-guest-agent
  - systemctl start qemu-guest-agent

write_files:
  - path: /etc/netplan/50-cloud-init.yaml
    content: |
      network:
        version: 2
        ethernets:
          ens3:
            dhcp4: true
EOF

cat > /tmp/$VM/meta-data <<EOF
instance-id: $VM
local-hostname: $VM
EOF

cloud-localds $STORAGE/$VM-seed.img \
/tmp/$VM/user-data \
/tmp/$VM/meta-data
}

create_vm(){
VM=$1
log "Creating VM $VM"
clone_disk $VM
create_cloud_init $VM

virt-install \
--name $VM \
--memory ${VM_RAM[$VM]} \
--vcpus ${VM_CPU[$VM]} \
--disk path=$STORAGE/$VM.qcow2,format=qcow2 \
--disk path=$STORAGE/$VM-seed.img,device=cdrom \
--network bridge=$BRIDGE,model=virtio \
--os-variant ubuntu22.04 \
--graphics none \
--import \
--noautoconsole
}

create_all_vms(){
for vm in "${VM_NAMES[@]}"
do
```

```
create_vm $vm
done

}

wait_for_ssh(){
IP=$1
log "Waiting for SSH $IP"
until ssh -o StrictHostKeyChecking=no -o
ConnectTimeout=3 ubuntu@$IP "echo ok" 2>/dev/null
do
sleep 5
done
log "$IP ready"
}

status_panel(){
echo "==== HOST ===="
uptime
echo ""
echo "==== VMS ===="
virsh list --all
echo ""
echo "==== STORAGE ===="
df -h
echo ""
echo "==== NETWORK ===="
ip a
}

menu(){

while true
do
clear
echo "================================"
echo " MINI PRIVATE CLOUD"
echo "================================"
echo "1 Detect host"
echo "2 Install host packages"
echo "3 Create VMs"
echo "4 Status panel"
echo ""
echo "20 Deploy base infrastructure"
echo ""
echo "0 Exit"
read -p "Option: " opt
```

```
case $opt in
1) detect_host ;;
2) install_host_packages ;;
3) create_all_vms ;;
4) status_panel ;;
20)
detect_host
install_host_packages
create_all_vms
;;
0) exit ;;
esac

pause
done
}

check_root
menu
```

```
admon@srv01:~$ sudo ./instala_sistema.sh
ERROR: No se encontró la clave pública en /home/
admon/.ssh/id_rsa.pub
Genera una con: ssh-keygen -t rsa -b 4096
admon@srv01:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/admon/.ssh/
id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/admon/.ssh/
id_rsa
Your public key has been saved in /home/admon/.ssh/
id_rsa.pub
The key fingerprint is:
SHA256:67pB....vdqOk admon@srv01
The key's randomart image is:
+---[RSA 4096]----+
|  .o oo.    . o   |
...
...
|       o         |
|      oo.        |
+----[SHA256]-----+
```

Ejecutamos hasta el punto 3 y modificamos dnsmasq

```
sudo systemctl status dnsmasq.service
sudo vi /etc/systemd/resolved.conf
sudo rm -f /etc/resolv.conf
sudo touch /etc/resolv.conf
sudo vi /etc/resolv.conf
sudo vi /etc/default/dnsmasq
```

Ahora configuramos (MÁS ADELANTE VER CÓMO TIENE QUE QUEDAR LA CONFIGURACIÓN DEL DNS SERVER)

```
port=53
domain-needed
bogus-priv
strict-order
listen-address=127.0.0.1
interface=br0 # escucha solo en bridge de VMs
bind-interfaces
no-dhcp-interface=
domain=informatica.iespoveda.com
cache-size=1000
expand-hosts                # usa hostname + domain
dhcp-range=172.16.6.20,172.16.6.100,12h  # rango DHCP
con lease de 12 horas
dhcp-option=3,172.16.6.1   # 3 -> default gateway para
las VMs
dhcp-option=6,172.16.6.5   # 6 -> default DNS para las
VMs (apunta a identity)
# Entradas fijas para VMs conocidas (opcional)
dhcp-host=identity,172.16.6.10
dhcp-host=database,172.16.6.20
dhcp-host=core,172.16.6.30
dhcp-host=nextcloud,172.16.6.40
dhcp-host=openproject,172.16.6.41
dhcp-host=fediverse,172.16.6.50
dhcp-host=observability,172.16.6.60
```

Reiniciamos sistema:

```
sudo systemctl restart systemd-networkd
sudo systemctl stop dnsmasq.service
sudo systemctl start dnsmasq.service
```

Y ahora continuamos con los pasos, el paso 3 y obtenemos:

```
admon@srv01:~$ sudo virsh list --all
 Id   Name              State
---------------------------------
 1    identity          running
 2    database          running
 3    core              running
 4    nextcloud         running
 5    openproject       running
 6    fediverse         running
 7    observability     running
 -    ubuntu2204-base   shut off
 -    ubuntu2404-base   shut off
```

Nos conectamos por ejemplo a una de las máquinas virtuales

```
admon@srv01:~$ ssh -i ~/.ssh/id_rsa ubuntu@172.16.6.20
The authenticity of host '172.16.6.20 (172.16.6.20)'
can't be established.
ED25519 key fingerprint is SHA256:8...Q.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/
[fingerprint])? yes
Warning: Permanently added '172.16.6.20' (ED25519) to
the list of known hosts.
Enter passphrase for key '/home/admon/.ssh/id_rsa':
Welcome to Ubuntu 24.04.4 LTS (GNU/Linux 6.8.0-101-
generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of lun 16 mar 2026 10:39:54 UTC

  System load:  1.36              Processes:
  156
  Usage of /:   45.3% of 9.75GB   Users logged in:
  0
  Memory usage: 7%                IPv4 address for
enp1s0: 172.16.6.20
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT
```

```
secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure
K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-
edge

El mantenimiento de seguridad expandido para
Applications está desactivado

Se pueden aplicar 0 actualizaciones de forma inmediata.

Active ESM Apps para recibir futuras actualizaciones de
seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»



The programs included with the Ubuntu system are free
software;
the exact distribution terms for each program are
described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by
applicable law.

To run a command as administrator (user "root"), use
"sudo <command>".
See "man sudo_root" for details.
```

O sea queda:

```
admon@srv01:~$ sudo virsh domiflist database
 Interface   Type     Source   Model    MAC
-------------------------------------------------------
 vnet1       bridge   br0      virtio
52:54:00:4f:af:b3

admon@srv01:~$ sudo virsh domifaddr database --source
agent
 Name       MAC address          Protocol     Address
-------------------------------------------------------
 lo         00:00:00:00:00:00    ipv4
127.0.0.1/8
```

```
 -           -                       ipv6          ::1/128
 enp1s0      52:54:00:4f:af:b3       ipv4
172.16.6.20/24
 -           -                       ipv6
fe80::5054:ff:fe4f:afb3/64
```

Certificados

Los instalamos en el servidor

```
admon@srv01:~$ sudo su -
[sudo] password for admon:
root@srv01:~# mkdir certs
root@srv01:~# cd certs
root@srv01:~/certs#  openssl genrsa -out iespoveda.key
4096
root@srv01:~/certs# openssl req -x509 -new -nodes -key
iespoveda.key -sha256 -days 3650 -out iespoveda.pem
You are about to be asked to enter information that will
be incorporated
into your certificate request.
What you are about to enter is what is called a
Distinguished Name or a DN.
There are quite a few fields but you can leave some
blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-
State]:Andalucia
Locality Name (eg, city) []:Guadix
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:IESPoveda
Organizational Unit Name (eg, section) []:Informatica
Common Name (e.g. server FQDN or YOUR name)
[]:informatica.iespoveda.com
Email Address []:admin@informatica.iespoveda.com

root@srv01:~/certs# cp iespoveda.pem /usr/local/share/
ca-certificates/iespoveda.crt
root@srv01:~/certs# update-ca-certificates
```

```
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does
not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

authentik

```
root@srv01:~/certs# vi authentik.conf
[req]
default_bits       = 2048
distinguished_name = authentik.iespoveda.com
req_extensions     = v3_req
prompt             = no

[authentik.iespoveda.com]
C  = ES
ST = Andalucia
L  = Guadix
O  = IESPoveda
CN = authentik.iespoveda.com

[v3_req]
subjectAltName = @alt_names

[alt_names]
DNS.1 = authentik.iespoveda.com
DNS.2 = auth.iespoveda.com
DNS.3 = sso.iespoveda.com
```

```
root@srv01:~/certs#  openssl req -new -nodes -out
authentik.csr -keyout authentik.key -config
authentik.conf
..+....+...+++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++*.................+..
+...................+...+...+....+.....+...++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++*....
+.........................+.....................
+........+....+..+....+...........+.+.........+........
+..........+......+.....+....+.....+......+......+.
+................+.....+....+.....+....+..+.......
+........+....+.....+.........+....+................+...
+..+.......+...........+...+....+..
```

```
+.......................+....+.....................+.
+......+...+.....+.+.....+.+.....+...+.+...........+...
+...+...................+.......+............+..
+..........+...+......+.........+...+..+.+..............
+......+......+.+.....+....++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++
..+.......+..+.................+.....++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++*.+.+..+...
+.+...+...........+........++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++*.+....+..+.........
+............+...+....+..................................
+........+.+...+...+...+......+......+..+.+.....+......
+.........+......+.+..............
+.........................+.+...........+...+.+...+..+...
+.......+.....+.+........+.+...+...+........+....
+........+.........+............+..+...+...+...
+..........+.........+............+.....+.+..+.+......
+...............+...+..+......+..........+.........+++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++
-----
root@srv01:~/certs# openssl req -in authentik.csr -noout
-text | grep -A1 "Subject Alternative Name"
                X509v3 Subject Alternative Name:
                    DNS:authentik.iespoveda.com,
DNS:auth.iespoveda.com, DNS:sso.iespoveda.com
```

```
root@srv01:~/certs# vi authentik.ext
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
DNS.1 = authentik.iespoveda.com
DNS.2 = auth.iespoveda.com
DNS.3 = sso.iespoveda.com
```

```
root@srv01:~/certs# openssl x509 -req   -in
authentik.csr   -CA iespoveda.pem   -CAkey iespoveda.key
  -CAcreateserial   -out authentik.crt   -days 825   -
sha256   -extfile authentik.ext
Certificate request self-signature ok
```

```
subject=C = ES, ST = Andalucia, L = Guadix, O =
IESPoveda, CN = authentik.iespoveda.com
```

ldap + authentik

Ahora instalamos ldap + authentik en la máquina "172.16.6.10":
Para ello nos conectamos vía ssh: `ssh -i ~/.ssh/id_rsa -o StrictHostKeyChecking=no ubuntu@172.16.6.10`

```
sudo dpkg-reconfigure slapd
```

Tras la instalación:

```
ubuntu@identity:~/authentik$ ldapsearch -x -LLL -H
ldap://localhost -b dc=iespoveda,dc=com
dn: dc=iespoveda,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: iespoveda
dc: iespoveda
```

ldapsearch -x -LLL -H ldap://localhost -b dc=iespoveda,dc=com

```
vi usuario_aortrei469.ldif
dn: cn=aortrei469,dc=iespoveda,dc=com
objectClass: inetOrgPerson
cn: arcadio
sn: ortega
mail: aortrei469@g.educaand.es
```

ldapadd -x -D "cn=admin,dc=iespoveda,dc=com" -W -f usuario_aortrei469.ldif

```
ubuntu@identity:~/authentik$ cat usuarios_educaand.ldif
dn: ou=geducaand,dc=iespoveda,dc=com
objectClass: top
objectClass: groupOfNames
```

```
cn: geducaand
member: cn=admin,dc=iespoveda,dc=com
```

```
ubuntu@identity:~/authentik$ ldapadd -x -D
"cn=admin,dc=iespoveda,dc=com" -W -f
usuarios_educaand.ldif
Enter LDAP Password:
adding new entry "ou=geducaand,dc=iespoveda,dc=com"
```

```
ubuntu@identity:~/authentik$ cat
agregar_aortrei469_a_geducaand.ldif
dn: ou=geducaand,dc=iespoveda,dc=com
changetype: modify
add: member
member: cn=aortrei469,ou=geducaand,dc=iespoveda,dc=com
```

```
ubuntu@identity:~/authentik$ ldapmodify -x -D
"cn=admin,dc=iespoveda,dc=com" -W -f
agregar_aortrei469_a_geducaand.ldif
Enter LDAP Password:
modifying entry "ou=geducaand,dc=iespoveda,dc=com"
```

Docker

```
sudo apt install docker.io docker-compose-v2 -y
ubuntu@identity:~/authentik$ sudo systemctl enable
docker
ubuntu@identity:~/authentik$ sudo systemctl start docker
ubuntu@identity:~/authentik$ docker --version
Docker version 28.2.2, build 28.2.2-0ubuntu1~24.04.1
ubuntu@identity:~/authentik$ docker compose version
Docker Compose version 2.37.1+ds1-0ubuntu2~24.04.1
```

Authentik

```
mkdir ~/authentik
cd ~/authentik
ubuntu@identity:~/authentik$ wget https://
docs.goauthentik.io/compose.yml
--2026-03-16 13:03:33--  https://docs.goauthentik.io/
compose.yml
```

```
Resolving docs.goauthentik.io (docs.goauthentik.io)...
104.26.10.86, 104.26.11.86, 172.67.74.168, ...
Connecting to docs.goauthentik.io (docs.goauthentik.io)|
104.26.10.86|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-yaml]
Saving to: ‘compose.yml’

compose.yml                                 [ <=>
                                                    ]
1,93K  --.-KB/s    in 0s

2026-03-16 13:03:34 (18,2 MB/s) - ‘compose.yml’ saved
[1975]

ubuntu@identity:~/authentik$ echo "PG_PASS=$(openssl
rand -base64 36 | tr -d '\n')" >> .env
ubuntu@identity:~/authentik$ echo "AUTHENTIK_SECRET_KEY=
$(openssl rand -base64 60 | tr -d '\n')" >> .env
ubuntu@identity:~/authentik$ echo
"AUTHENTIK_ERROR_REPORTING__ENABLED=true" >> .env
ubuntu@identity:~/authentik$ vi docker-compose.yml
services:
  postgresql:
    env_file:
    - .env
    environment:
      POSTGRES_DB: ${PG_DB:-authentik}
      POSTGRES_PASSWORD: ${PG_PASS:?database password
required}
      POSTGRES_USER: ${PG_USER:-authentik}
    healthcheck:
      interval: 30s
      retries: 5
      start_period: 20s
      test:
      - CMD-SHELL
      - pg_isready -d $${POSTGRES_DB} -U $
${POSTGRES_USER}
      timeout: 5s
    image: docker.io/library/postgres:16-alpine
    restart: unless-stopped
    volumes:
    - database:/var/lib/postgresql/data
  server:
    command: server
    depends_on:
```

```
      postgresql:
        condition: service_healthy
    env_file:
    - .env
    environment:
      AUTHENTIK_POSTGRESQL__HOST: postgresql
      AUTHENTIK_POSTGRESQL__NAME: ${PG_DB:-authentik}
      AUTHENTIK_POSTGRESQL__PASSWORD: ${PG_PASS}
      AUTHENTIK_POSTGRESQL__USER: ${PG_USER:-authentik}
      AUTHENTIK_SECRET_KEY: ${AUTHENTIK_SECRET_KEY:?
secret key required}
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/
server}:${AUTHENTIK_TAG:-2025.12.1}
    ports:
    - ${COMPOSE_PORT_HTTP:-9000}:9000
    - ${COMPOSE_PORT_HTTPS:-9443}:9443
    restart: unless-stopped
    volumes:
    - ./data:/data
    - ./custom-templates:/templates
    - ./certs:/certs
  worker:
    command: worker
    depends_on:
      postgresql:
        condition: service_healthy
    env_file:
    - .env
    environment:
      AUTHENTIK_POSTGRESQL__HOST: postgresql
      AUTHENTIK_POSTGRESQL__NAME: ${PG_DB:-authentik}
      AUTHENTIK_POSTGRESQL__PASSWORD: ${PG_PASS}
      AUTHENTIK_POSTGRESQL__USER: ${PG_USER:-authentik}
      AUTHENTIK_SECRET_KEY: ${AUTHENTIK_SECRET_KEY:?
secret key required}
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/
server}:${AUTHENTIK_TAG:-2025.12.1}
    restart: unless-stopped
    user: root
    volumes:
    - /var/run/docker.sock:/var/run/docker.sock
    - ./data:/data
    - ./certs:/certs
    - ./custom-templates:/templates
volumes:
  database:
    driver: local
```

```
ubuntu@identity:~/authentik$ sudo docker compose pull
WARN[0000] Found multiple config files with supported
names: /home/ubuntu/authentik/compose.yml, /home/ubuntu/
authentik/docker-compose.yml
WARN[0000] Using /home/ubuntu/authentik/compose.yml
[+] Pulling 40/40
 ✔ server Skipped - Image is already being pulled by
worker
                                0.0s
 ✔ postgresql Pulled

                            22.3s
 ✔ worker Pulled
```

```
ubuntu@identity:~/authentik$ sudo apt install
phpldapadmin
...
...
Se instalarán los siguientes paquetes NUEVOS:
  apache2 apache2-bin apache2-data apache2-utils
libapache2-mod-php8.3 libapr1t64
  libaprutil1-dbd-sqlite3 libaprutil1-ldap
libaprutil1t64 liblua5.4-0 php php-common php-ldap php-
xml
  php8.3 php8.3-cli php8.3-common php8.3-ldap php8.3-
opcache php8.3-readline php8.3-xml phpldapadmin
  ssl-cert
...
...
```

```
ubuntu@identity:~/authentik$ sudo ss -tulipn
Netid State  Recv-Q Send-Q      Local Address:Port
Peer Address:Port Process
udp   UNCONN 0      0              127.0.0.54:53
  0.0.0.0:* users:(('systemd-resolve',pid=547,fd=16))
udp   UNCONN 0      0           127.0.0.53%lo:53
  0.0.0.0:* users:(('systemd-resolve',pid=547,fd=14))
udp   UNCONN 0      0      172.16.6.10%enp1s0:68
  0.0.0.0:* users:(('systemd-network',pid=706,fd=21))
tcp   LISTEN 0      4096           127.0.0.1:46625
  0.0.0.0:* users:(('containerd',pid=1603,fd=9))
         cubic cwnd:10
tcp   LISTEN 0      2048              0.0.0.0:389
  0.0.0.0:* users:(('slapd',pid=9669,fd=8))
```

```
         cubic cwnd:10
tcp   LISTEN 0      4096              0.0.0.0:22
  0.0.0.0:* users:(('sshd',pid=2472,fd=3),
('systemd',pid=1,fd=112))
         cubic cwnd:10
tcp   LISTEN 0      4096           127.0.0.54:53
  0.0.0.0:* users:(('systemd-resolve',pid=547,fd=17))
         cubic cwnd:10
tcp   LISTEN 0      4096        127.0.0.53%lo:53
  0.0.0.0:* users:(('systemd-resolve',pid=547,fd=15))
         cubic cwnd:10
tcp   LISTEN 0      2048                  [::]:389
     [::]:* users:(('slapd',pid=9669,fd=9))
         cubic cwnd:10
tcp   LISTEN 0      4096                  [::]:22
     [::]:* users:(('sshd',pid=2472,fd=4),
('systemd',pid=1,fd=116))
         cubic cwnd:10
tcp   LISTEN 0      511                      *:80
        *:* users:(('apache2',pid=19261,fd=4),
('apache2',pid=19260,fd=4),('apache2',pid=19259,fd=4),
('apache2',pid=19258,fd=4),('apache2',pid=19257,fd=4),
('apache2',pid=19254,fd=4))
```

```
ubuntu@identity:~/authentik$ ls /etc/apache2/conf-
enabled/phpldapadmin.conf
/etc/apache2/conf-enabled/phpldapadmin.conf
```

Nos traemos los certificados generados para identity

```
ubuntu@identity:~/authentik$ sftp admon@172.16.6.5
The authenticity of host '172.16.6.5 (172.16.6.5)' can't
be established.
ED25519 key fingerprint is SHA256:5...w.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/
[fingerprint])? yes
Warning: Permanently added '172.16.6.5' (ED25519) to the
list of known hosts.
admon@172.16.6.5's password:
Connected to 172.16.6.5.
sftp> cd /tmp
sftp> mget authentik.*
Fetching /tmp/authentik.crt to authentik.crt
authentik.crt
```

```
100% 1927     1.1MB/s   00:00
Fetching /tmp/authentik.key to authentik.key
authentik.key

100% 1704     1.1MB/s   00:00
sftp> mget /tmp/iespoveda.pem
Fetching /tmp/iespoveda.pem to iespoveda.pem
iespoveda.pem

100% 2232     1.6MB/s   00:00
sftp> bye
```

Ahora lo instalamos para nginx reverse, authentik y para ser conocido en el server

```
ubuntu@identity:~/authentik$ sudo cp authentik.crt
authentik.key iespoveda.pem /etc/ssl/authentik/
ubuntu@identity:~/authentik$ mkdir certs
ubuntu@identity:~/authentik$ cd certs/
ubuntu@identity:~/authentik/certs$ sudo mv ../
authentik.* .
ubuntu@identity:~/authentik/certs$ sudo mv ../
iespoveda.pem .
ubuntu@identity:~/authentik/certs$ ls -la
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 mar 17 12:53 .
drwxrwxr-x 3 ubuntu ubuntu 4096 mar 17 12:53 ..
-rw-r--r-- 1 ubuntu ubuntu 1927 mar 17 12:51
authentik.crt
-rw-r--r-- 1 ubuntu ubuntu 1704 mar 17 12:51
authentik.key
-rw-r--r-- 1 ubuntu ubuntu 2232 mar 17 12:51
iespoveda.pem
ubuntu@identity:~/authentik/certs$ sudo chmod 600
authentik.key
ubuntu@identity:~/authentik/certs$ sudo chmod 600 /etc/
ssl/authentik/authentik.key
ubuntu@identity:~/authentik/certs$ sudo cp iespoveda.pem
/usr/local/share/ca-certificates/iespoveda.crt
ubuntu@identity:~/authentik/certs$ sudo update-ca-
certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does
not contain exactly one certificate or CRL
1 added, 0 removed; done.
```

```
Running hooks in /etc/ca-certificates/update.d...
done.
```

Verificamos:

```
ubuntu@identity:~/authentik/certs$ openssl verify -
CAfile iespoveda.pem authentik.crt
authentik.crt: OK
```

Instalamos nginx:

```
sudo apt install nginx -y
```

```
ubuntu@identity:~/authentik$ sudo docker compose up -d
WARN[0000] Found multiple config files with supported
names: /home/ubuntu/authentik/compose.yml, /home/ubuntu/
authentik/docker-compose.yml
WARN[0000] Using /home/ubuntu/authentik/compose.yml
[+] Running 5/5
 ✔ Network authentik_default      Created

                            0.2s
 ✔ Volume "authentik_database"     Created

                             0.0s
 ✔ Container authentik-postgresql-1  Healthy

                               6.5s
 ✔ Container authentik-server-1      Started

                               6.7s
 ✔ Container authentik-worker-1      Started
```

```
ubuntu@identity:~/authentik/certs$ sudo cat /etc/nginx/
sites-available/authentik
server {
    listen 443 ssl;
    server_name authentik.iespoveda.com;

    ssl_certificate /etc/ssl/authentik/authentik.crt;
    ssl_certificate_key /etc/ssl/authentik/
authentik.key;
```

```
    ssl_protocols TLSv1.2 TLSv1.3;

    location / {
        proxy_pass http://127.0.0.1:9000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

```
ubuntu@identity:/etc/nginx$ cd sites-enabled/
ubuntu@identity:/etc/nginx/sites-enabled$ ls
authentik  default
ubuntu@identity:/etc/nginx/sites-enabled$ sudo rm -f
default
```

Ahora verificamos todo:

```
ubuntu@identity:~/authentik$ sudo systemctl stop nginx
ubuntu@identity:~/authentik$ sudo systemctl start nginx
ubuntu@identity:~/authentik$ sudo systemctl status nginx
● nginx.service - A high performance web server and a
reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/
nginx.service; enabled; preset: enabled)
     Active: active (running) since Tue 2026-03-17
13:07:31 UTC; 3s ago
       Docs: man:nginx(8)
    Process: 5277 ExecStartPre=/usr/sbin/nginx -t -q -g
daemon on; master_process on; (code=exited, status=0/
SUCCESS)
    Process: 5279 ExecStart=/usr/sbin/nginx -g daemon
on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 5280 (nginx)
      Tasks: 3 (limit: 3391)
     Memory: 2.7M (peak: 2.9M)
        CPU: 32ms
     CGroup: /system.slice/nginx.service
             5280 "nginx: master process /usr/sbin/
nginx -g daemon on; master_process on;"
             5281 "nginx: worker process"
             5282 "nginx: worker process"
```

```
mar 17 13:07:31 identity systemd[1]: Starting
nginx.service - A high performance web server and a
reverse proxy server...
mar 17 13:07:31 identity systemd[1]: Started
nginx.service - A high performance web server and a
reverse proxy server.
ubuntu@identity:~/authentik$ curl https://
authentik.iespoveda.com/.well-known/openid-configuration
authentik starting
```

IMPORTANTE Primera conexión:

```
https://authentik.iespoveda.com/if/flow/initial-setup/
```

Usuarios POSIX y Proxy authentik

```
#!/bin/bash

set -e

DOMAIN="iespoveda.com"
DC1="iespoveda"
DC2="com"
BASE_DN="dc=$DC1,dc=$DC2"
ADMIN_DN="cn=admin,$BASE_DN"

LDAP_PASS="l...l"
PROXY_PASS="l...l"
USER_PASS="l...l"

USER_NAME="arcadio"
USER_UID="1005"
GROUP_NAME="usuarios"
GROUP_GID="100"

echo "=== Creando estructura base ==="

cat > base.ldif <<EOF
dn: ou=usuarios,$BASE_DN
```

```
objectClass: organizationalUnit
ou: usuarios

dn: ou=grupos,$BASE_DN
objectClass: organizationalUnit
ou: grupos
EOF

ldapadd -x -D "$ADMIN_DN" -w $LDAP_PASS -f base.ldif

echo "=== Creando grupo POSIX ==="

cat > grupo.ldif <<EOF
dn: cn=$GROUP_NAME,ou=grupos,$BASE_DN
objectClass: posixGroup
cn: $GROUP_NAME
gidNumber: $GROUP_GID
EOF

ldapadd -x -D "$ADMIN_DN" -w $LDAP_PASS -f grupo.ldif

echo "=== Creando usuario POSIX ==="

HASH_USER=$(slappasswd -s $USER_PASS)

cat > usuario.ldif <<EOF
dn: uid=$USER_NAME,ou=usuarios,$BASE_DN
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: $USER_NAME
sn: $USER_NAME
uid: $USER_NAME
uidNumber: $USER_UID
gidNumber: $GROUP_GID
homeDirectory: /home/$USER_NAME
loginShell: /bin/bash
userPassword: $HASH_USER
EOF

ldapadd -x -D "$ADMIN_DN" -w $LDAP_PASS -f usuario.ldif

echo "=== Añadiendo usuario al grupo ==="

cat > addmember.ldif <<EOF
dn: cn=$GROUP_NAME,ou=grupos,$BASE_DN
changetype: modify
```

```
add: memberUid
memberUid: $USER_NAME
EOF

ldapmodify -x -D "$ADMIN_DN" -w $LDAP_PASS -f
addmember.ldif

echo "=== Creando usuario proxy para Authentik ==="

HASH_PROXY=$(slappasswd -s $PROXY_PASS)

cat > proxy.ldif <<EOF
dn: cn=proxyuser,$BASE_DN
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: proxyuser
description: Proxy user for LDAP clients
userPassword: $HASH_PROXY
EOF

ldapadd -x -D "$ADMIN_DN" -w $LDAP_PASS -f proxy.ldif

echo "=== Verificación ==="
ldapsearch -x -b "$BASE_DN" uid=$USER_NAME
ldapsearch -x -b "$BASE_DN" cn=$GROUP_NAME

echo ""
echo "========================================="
echo " LDAP listo para Authentik"
echo " Base DN: $BASE_DN"
echo " Admin:   $ADMIN_DN"
echo " Proxy:   cn=proxyuser,$BASE_DN"
echo " Usuario: $USER_NAME"
echo "========================================="
```

Una vez ejecutado ya lo tenemos. Verificamos

```
ubuntu@identity:~$ ldapsearch -x -D
"cn=proxyuser,dc=iespoveda,dc=com" -W -b
"dc=iespoveda,dc=com" "(objectClass=posixAccount)"
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <dc=iespoveda,dc=com> with scope subtree
# filter: (objectClass=posixAccount)
```

```
# requesting: ALL
#

# arcadio, usuarios, iespoveda.com
dn: uid=arcadio,ou=usuarios,dc=iespoveda,dc=com
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: arcadio
sn: arcadio
uid: arcadio
uidNumber: 1005
gidNumber: 100
homeDirectory: /home/arcadio
loginShell: /bin/bash

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

ldap ssl

En el servidor Definimos:

```
root@srv01:~/certs# cat ldap.conf
[req]
default_bits       = 2048
distinguished_name = ldap.iespoveda.com
req_extensions     = v3_req
prompt             = no

[ldap.iespoveda.com]
C  = ES
ST = Andalucia
L  = Guadix
O  = IESPoveda
CN = ldap.iespoveda.com

[v3_req]
subjectAltName = @alt_names

[alt_names]
```

```
DNS.1 = ldap.iespoveda.com
root@srv01:~/certs# cat ldap.ext
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
DNS.1 = ldap.iespoveda.com
```

```
root@srv01:~/certs# openssl req -new -nodes -out
ldap.csr -keyout ldap.key -config ldap.conf
......++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++*....++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++*..+.....................+.
+........+......+.......................+.....+...+....
+.....+...+...+.+...................+..+.+............
+........+....+.............+.............+.+.....+.
+..+.+.........+..+.+.........+.....++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++
......+.+............+.....+.++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++*..+.+.....+....
+...+......+.....+......+...+...+.......+..+......+....
+......+........+....+...+..+.++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++*........
+.........+.+.........++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++
-----
root@srv01:~/certs# openssl req -in ldap.csr -noout -
text | grep -A1 "Subject Alternative Name"
                X509v3 Subject Alternative Name:
                    DNS:ldap.iespoveda.com
root@srv01:~/certs# openssl x509 -req -in ldap.csr -CA
iespoveda.pem -CAkey iespoveda.key -CAcreateserial -out
ldap.crt -days 825 -sha256 -extfile ldap.ext
Certificate request self-signature ok
subject=C = ES, ST = Andalucia, L = Guadix, O =
IESPoveda, CN = ldap.iespoveda.com
```

En el servidor de ldap nos queda:

```
buntu@identity:/etc/ldap/sasl2$ ls -la
```

```
total 20
drwxr-xr-x 2 root root 4096 mar 18 09:43 .
drwxr-xr-x 5 root root 4096 mar 17 12:30 ..
-rw-r--r-- 1 root root 2232 mar 18 09:43 iespoveda.crt
-rw-r--r-- 1 root root 1862 mar 18 09:40 ldap.crt
-rw------- 1 root root 1704 mar 18 09:40 ldap.key
sudo chown openldap:openldap /etc/ldap/sasl2/*.crt
sudo chown openldap:openldap /etc/ldap/sasl2/*.key
sudo chmod 644 /etc/ldap/sasl2/*.crt
sudo chmod 600 /etc/ldap/sasl2/*.key
```

Ahora lo definimos:

```
ubuntu@identity:~/ldap_certs$ cat certificados.ldif
dn: cn=config
changetype: modify
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/sasl2/ldap.crt
-
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/sasl2/ldap.key
-
replace: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ldap/sasl2/iespoveda.crt
```

Ahora ejecutamos:

```
ubuntu@identity:~/ldap_certs$ sudo ldapmodify -Y
EXTERNAL -H ldapi:/// -f certificados.ldif
SASL/EXTERNAL authentication started
SASL username:
gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"
```

Además necesitamos:

```
sudo grep SLAPD_SERVICES /etc/default/slapd
# SLAPD_SERVICES="ldap://127.0.0.1:389/ ldaps:///
ldapi:///"
SLAPD_SERVICES="ldap:/// ldapi:/// ldaps:///"
```

Reiniciamos el servicio

```
sudo systemctl restart slapd
```

Probamos (Necesitamos la entrada en dns server, en mi caso `cname=ldap.iespoveda.com,identity`):

```
ubuntu@identity:~/ldap_certs$ ldapsearch -x -H ldaps://
ldap.iespoveda.com -b "dc=iespoveda,dc=com" | head
# extended LDIF
#
# LDAPv3
# base <dc=iespoveda,dc=com> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# iespoveda.com
dn: dc=iespoveda,dc=com
```

Es conveniente también cambiar:

```
ubuntu@identity:~/ldap_certs$ grep TLS /etc/ldap/
ldap.conf
# TLS certificates (needed for GnuTLS)
TLS_CACERT      /etc/ssl/certs/ca-certificates.crt
```

para garantizar la confianza permanente

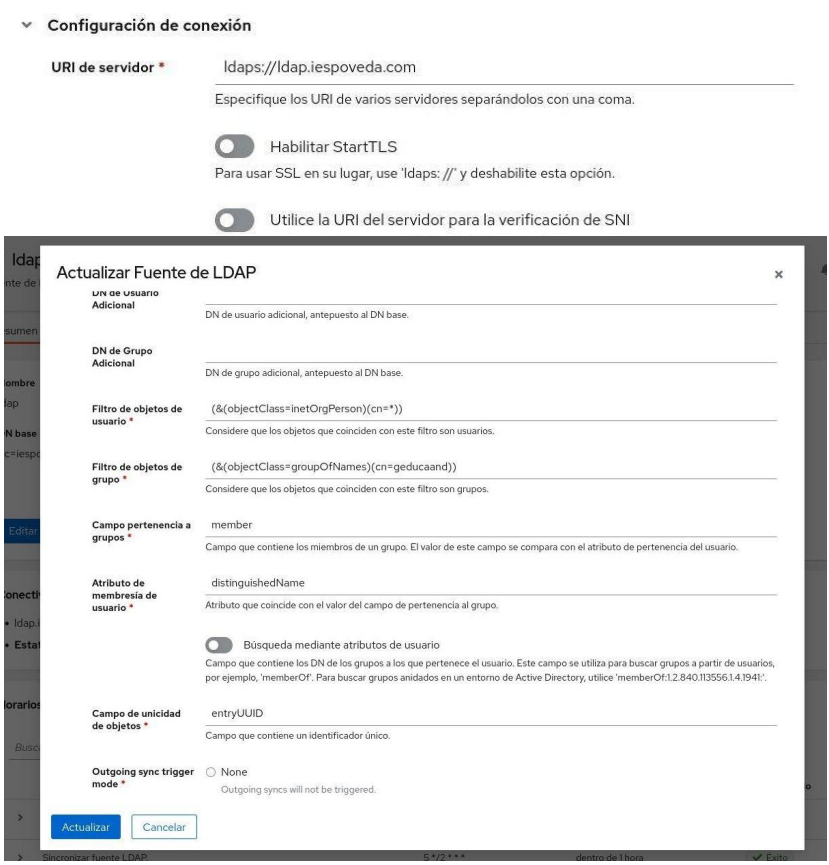
```
TLS_CACERT      /etc/ldap/sasl2/iespoveda.crt
```

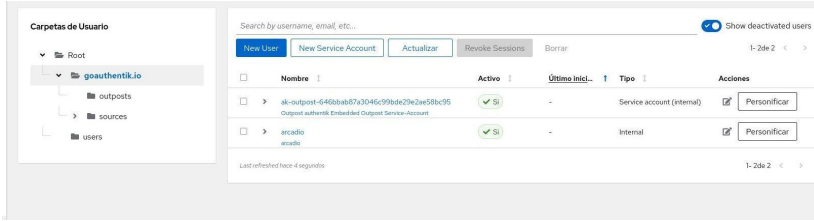
Authentik con ldap

Tuve probemas como estaba definido el grupo **Sólo quiero los de geducaand** no los de posix. Así:

```
ubuntu@identity:~$ cat
```

```
modifica_aortrei469_geducaand.ldif
dn: ou=geducaand,dc=iespoveda,dc=com
changetype: modify
delete: member
member: cn=aortrei469,ou=geducaand,dc=iespoveda,dc=com
-
add: member
member: cn=aortrei469,dc=iespoveda,dc=com
ubuntu@identity:~$ ldapmodify -x -D
"cn=admin,dc=iespoveda,dc=com" -W -f
modifica_aortrei469_geducaand.ldif
Enter LDAP Password:
modifying entry "ou=geducaand,dc=iespoveda,dc=com"
```





Configuración incial Máquinas virtuales

```
sudo virsh list --all
sudo virsh start nextcloud
sudo virsh domifaddr identity --source agent
sudo virsh domiflist nextcloud
sudo virsh shutdown nextcloud
sudo guestmount -a /vm/libvirt/images/nextcloud.qcow2 -i
/mnt
sudo vi /mnt/etc/netplan/50-cloud-init.yaml
```

Ponemos bien el nombre de la red que será enp1s0, casi seguro

```
sudo guestunmount  /mnt
sudo virsh start nextcloud
sudo virsh domifaddr identity --source agent
```

Actualizamos el dnsmasq con la ethernet de nextcloud (VER AL FINAL EJEMPLO DNSMASQ COMO QUEDA):

```
sudo vi /etc/dnsmasq.conf
sudo grep nextcloud /etc/dnsmasq.conf
dhcp-
host=52:54:00:95:0d:41,172.16.6.40,nextcloud,infinite
```

Reiniciamos dnsmas si hiciera falta así como la máquina virtual

```
sudo virsh reboot nextcloud
```

Quedando:

```

```

```
admon@srv01:~$ sudo virsh domiflist nextcloud
 Interface   Type      Source   Model    MAC
-------------------------------------------------------------
 vnet3       bridge    br0      virtio
52:54:00:95:0d:41

admon@srv01:~$ sudo virsh domifaddr nextcloud --source
agent
 Name        MAC address          Protocol     Address
-------------------------------------------------------------
 lo          00:00:00:00:00:00    ipv4
127.0.0.1/8
 -           -                    ipv6         ::1/128
 enp1s0      52:54:00:95:0d:41    ipv4
172.16.6.40/24
 -           -                    ipv6
fe80::5054:ff:fe95:d41/64
 docker0     f2:a8:74:da:ae:84    ipv4
172.17.0.1/16
```

Ahora nos conectamos:

```
admon@srv01:~$ ssh -i ~/.ssh/id_rsa -o
StrictHostKeyChecking=no ubuntu@172.16.6.40
```

Nextcloud

```
ubuntu@nextcloud:~$ sudo mysql
Welcome to the MariaDB monitor.  Commands end with ; or
\g.
Your MariaDB connection id is 31
Server version: 10.11.14-MariaDB-0ubuntu0.24.04.1 Ubuntu
24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab
and others.

Type 'help;' or '\h' for help. Type '\c' to clear the
current input statement.
```

```
MariaDB [(none)]> CREATE USER 'admin'@'localhost'
IDENTIFIED BY '...password...';
Query OK, 0 rows affected (0,012 sec)

MariaDB [(none)]> CREATE DATABASE IF NOT EXISTS
nextcloud CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci;
Query OK, 1 row affected (0,001 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nextcloud.* TO
'admin'@'localhost';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> \q
Bye
ubuntu@nextcloud:~$ sudo cp -r nextcloud /var/www
ubuntu@nextcloud:~$ sudo chown -R www-data:www-data /
var/www/nextcloud
ubuntu@nextcloud:~$ sudo a2enmod proxy proxy_fcgi
setenvif headers rewrite ssl
Enabling module proxy.
Considering dependency proxy for proxy_fcgi:
Module proxy already enabled
Enabling module proxy_fcgi.
Module setenvif already enabled
Enabling module headers.
Enabling module rewrite.
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to
configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@nextcloud:~$ php -v
PHP 8.3.6 (cli) (built: Jan 27 2026 03:09:47) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.3.6, Copyright (c) Zend Technologies
    with Zend OPcache v8.3.6, Copyright (c), by Zend
Technologies
ubuntu@nextcloud:~$ sudo apt install php8.3-fpm php8.3-
cli php8.3-gd php8.3-curl php8.3-xml php8.3-zip php8.3-
```

```
mbstring php8.3-intl php8.3-bz2 php8.3-imagick php8.3-
mysql
ubuntu@nextcloud:~$ sudo systemctl enable php8.3-fpm
Synchronizing state of php8.3-fpm.service with SysV
service script with /usr/lib/systemd/systemd-sysv-
install.
Executing: /usr/lib/systemd/systemd-sysv-install enable
php8.3-fpm
ubuntu@nextcloud:~$ sudo systemctl start php8.3-fpm
ubuntu@nextcloud:~$ sudo systemctl status php8.3-fpm
● php8.3-fpm.service - The PHP 8.3 FastCGI Process
Manager
     Loaded: loaded (/usr/lib/systemd/system/php8.3-
fpm.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-18
11:35:11 UTC; 26s ago
       Docs: man:php-fpm8.3(8)
   Main PID: 18092 (php-fpm8.3)
     Status: "Processes active: 0, idle: 2, Requests: 0,
slow: 0, Traffic: 0req/sec"
      Tasks: 3 (limit: 4601)
     Memory: 11.8M (peak: 12.6M)
        CPU: 115ms
     CGroup: /system.slice/php8.3-fpm.service
             18092 "php-fpm: master process (/etc/
php/8.3/fpm/php-fpm.conf)"
             18093 "php-fpm: pool www"
             18094 "php-fpm: pool www"

mar 18 11:35:10 nextcloud systemd[1]: Starting php8.3-
fpm.service - The PHP 8.3 FastCGI Process Manager...
mar 18 11:35:11 nextcloud systemd[1]: Started php8.3-
fpm.service - The PHP 8.3 FastCGI Process Manager.
ubuntu@nextcloud:~$ ls /run/php/
php8.3-fpm.pid  php8.3-fpm.sock  php-fpm.sock
ubuntu@nextcloud:~$ sudo mkdir /etc/ssl/nextcloud
ubuntu@nextcloud:~$ sudo mv nextcloud.key /etc/ssl/
nextcloud/.
ubuntu@nextcloud:~$ sudo mv nextcloud.crt /etc/ssl/
nextcloud/.
ubuntu@nextcloud:~$ sudo mv nextcloud.csr /etc/ssl/
nextcloud/.
ubuntu@nextcloud:~$ sudo chmod 600 /etc/ssl/nextcloud/
nextcloud.key
ubuntu@nextcloud:~$ sudo ls -la /etc/ssl/nextcloud/
nextcloud.key
-rw------- 1 ubuntu ubuntu 1704 mar 18 11:42 /etc/ssl/
```

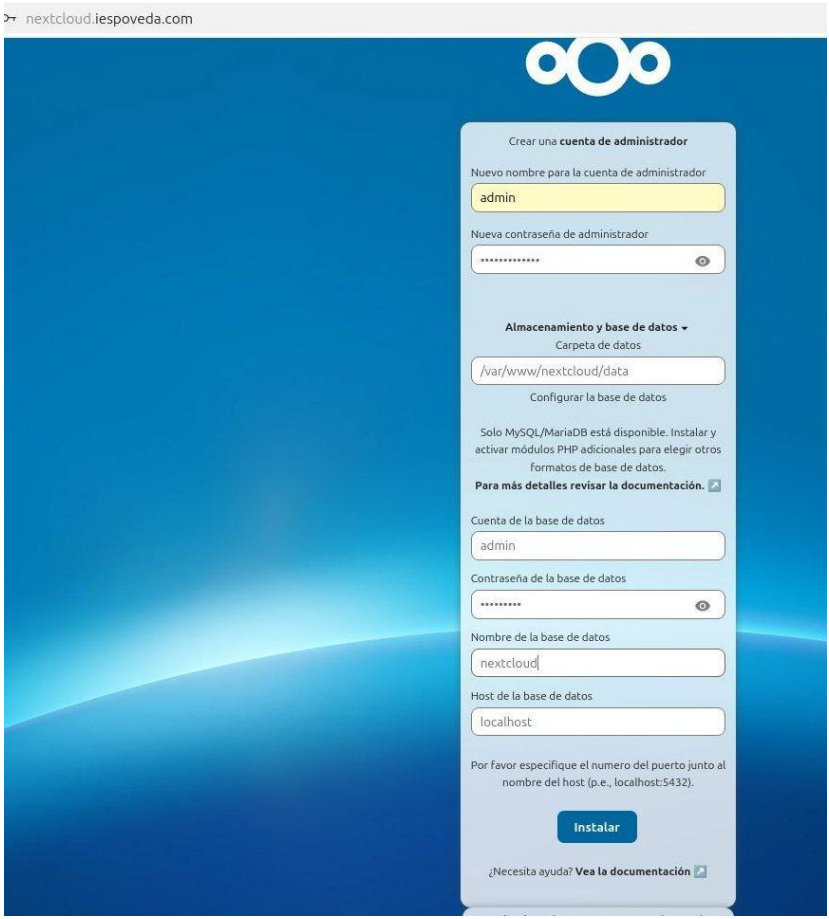
```
nextcloud/nextcloud.key
ubuntu@nextcloud:~$ sudo mv iespoveda.pem /usr/local/
share/ca-certificates/iespoveda.crt
ubuntu@nextcloud:~$ sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does
not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

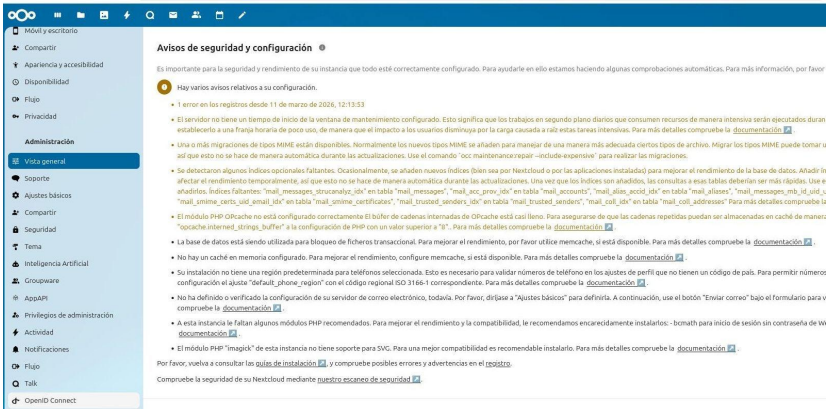
```
ubuntu@nextcloud:~$ sudo vi /etc/apache2/sites-
available/nextcloud.conf
<VirtualHost *:443>
    ServerName cloud.iespoveda.com

    ErrorLog ${APACHE_LOG_DIR}/nextcloud_error.log
    CustomLog ${APACHE_LOG_DIR}/nextcloud_access.log
combined

    SSLEngine on
    SSLCertificateFile      /etc/ssl/nextcloud/
nextcloud.crt
    SSLCertificateKeyFile   /etc/ssl/nextcloud/
nextcloud.key

    DocumentRoot /var/www/nextcloud

    <Directory /var/www/nextcloud/>
        Require all granted
        AllowOverride All
        Options FollowSymLinks MultiViews

        <IfModule mod_dav.c>
            Dav off
        </IfModule>
    </Directory>

    <IfModule mod_proxy_fcgi.c>
        <FilesMatch \.php$>
            SetHandler "proxy:unix:/run/php/php8.3-
fpm.sock|fcgi://localhost/"
        </FilesMatch>
    </IfModule>

    <FilesMatch \.php$>
```

```
        SetEnvIf Authorization "(.*)"
HTTP_AUTHORIZATION=$1
    </FilesMatch>

    # Seguridad recomendada
    <IfModule mod_headers.c>
        Header always set Strict-Transport-Security
"max-age=15552000; includeSubDomains"
        Header always set Referrer-Policy "no-referrer"
        Header always set X-Content-Type-Options
"nosniff"
        Header always set X-Frame-Options "SAMEORIGIN"
    </IfModule>

</VirtualHost>
```

```
ubuntu@nextcloud:~$ sudo rm -f /etc/apache2/sites-
enabled/000-default.conf
ubuntu@nextcloud:~$ cd /etc/apache2/sites-enabled/
ubuntu@nextcloud:/etc/apache2/sites-enabled$ sudo ln -s
../sites-available/
000-default.conf  default-ssl.conf  nextcloud.conf
ubuntu@nextcloud:/etc/apache2/sites-enabled$ sudo ln -s
../sites-available/nextcloud.conf
ubuntu@nextcloud:/etc/apache2/sites-enabled$ ls -la
total 8
drwxr-xr-x 2 root root 4096 mar 18 11:47 .
drwxr-xr-x 8 root root 4096 mar 18 11:27 ..
lrwxrwxrwx 1 root root   33 mar 18 11:47 nextcloud.conf
-> ../sites-available/nextcloud.conf
ubuntu@nextcloud:/etc/apache2/sites-enabled$  apachectl
configtest
Syntax OK
ubuntu@nextcloud:/etc/apache2/sites-enabled$ sudo
systemctl status apache2.service
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/
apache2.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-18
11:35:10 UTC; 12min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 18068 (apache2)
      Tasks: 6 (limit: 4601)
     Memory: 16.3M (peak: 16.4M)
        CPU: 220ms
```

```
     CGroup: /system.slice/apache2.service
             18068 /usr/sbin/apache2 -k start
             18073 /usr/sbin/apache2 -k start
             18074 /usr/sbin/apache2 -k start
             18075 /usr/sbin/apache2 -k start
             18076 /usr/sbin/apache2 -k start
             18077 /usr/sbin/apache2 -k start

mar 18 11:35:10 nextcloud systemd[1]: Starting
apache2.service - The Apache HTTP Server...
mar 18 11:35:10 nextcloud systemd[1]: Started
apache2.service - The Apache HTTP Server.
ubuntu@nextcloud:/etc/apache2/sites-enabled$ sudo
systemctl restart apache2.service
```





Tenemos que cambiar el límite de la memoria

```
grep memory_limit /etc/php/8.3/fpm/php.ini
memory_limit = 512M
sudo systemctl restart php8.3-fpm
sudo systemctl restart apache2
```

Integración Authentik + Nextcloud con OIDC

Seguimos los pasos de nuestra página web:
Ahora en nextcloud server

```
sudo -u www-data php /var/www/nextcloud/occ app:enable
user_oidc
user_oidc 8.6.1 enabled
```

Seguimos las indicaciones de la página web:

Configuración OIDC en Nextcloud

Instalar app:

```
sudo -u www-data php /var/www/nextcloud/occ app:enable
user_oidc
```

En la interfaz web:

- Discovery URL:

```
https://authentik.iespoveda.com/application/o/
nextcloud/.well-known/openid-configuration
```

- Client ID: (desde Authentik)
- Client Secret: (desde Authentik)
- Activar:
  - ○ Auto-create users
  - ○ Update user data
- En config.php:

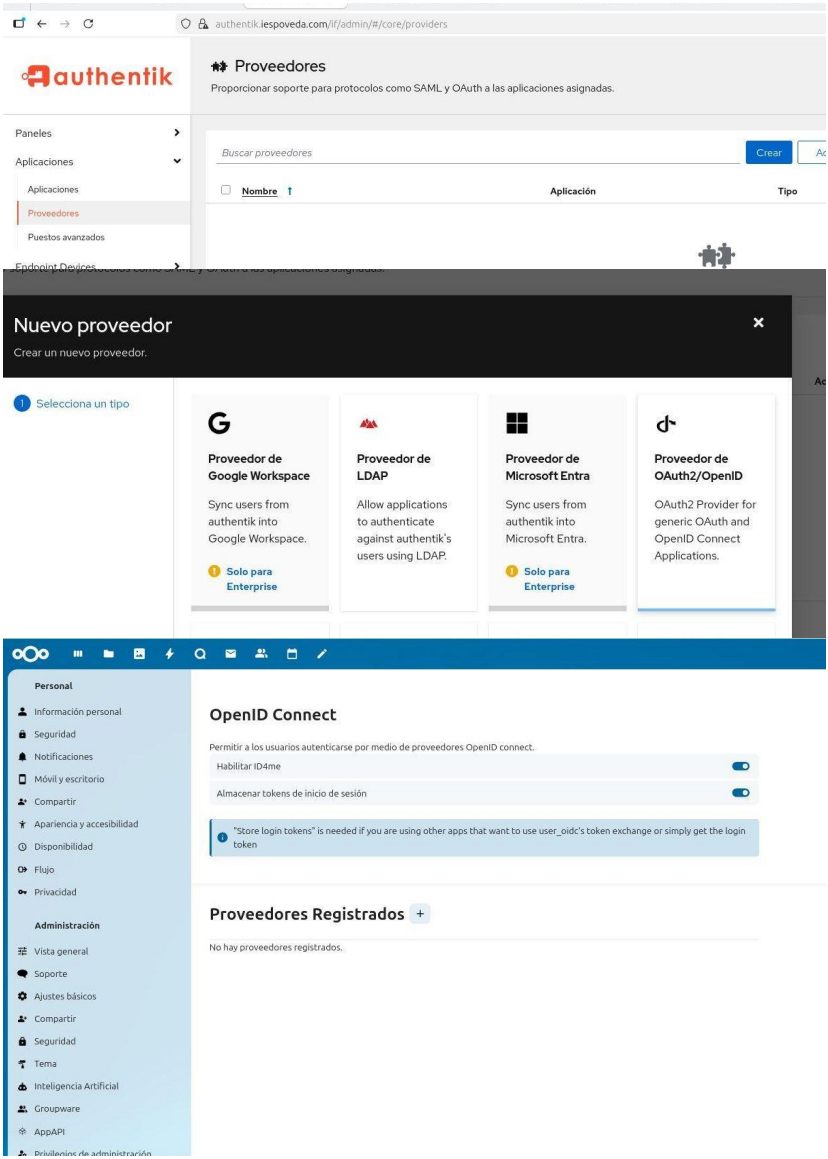
```
'allow_local_remote_servers' => true,
```

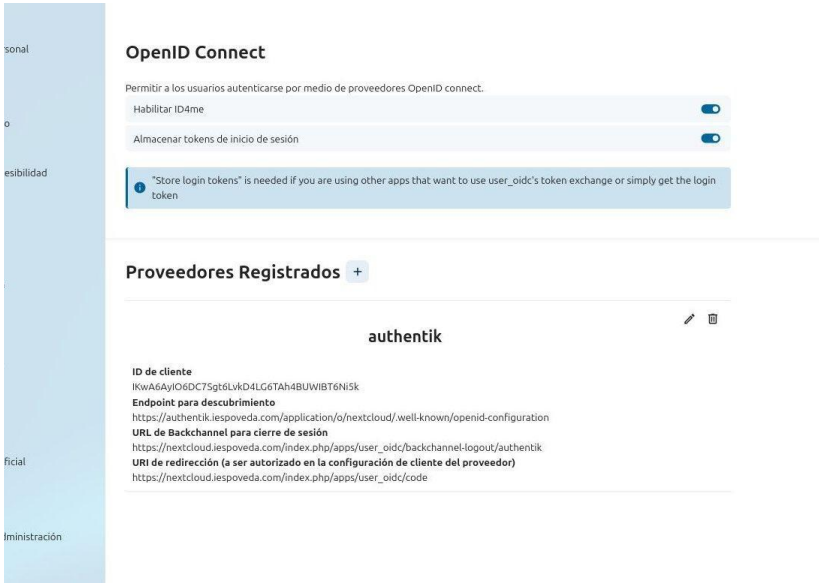
Quedando:

```
ubuntu@nextcloud:/tmp$ sudo cat  /var/www/
nextcloud/config/config.php
<?php
$CONFIG = array (
...
...
  'installed' => true,
  'allow_local_remote_servers' => true,
  'overwriteprotocol' => 'https'
);
```

- Reinicio:

```
sudo systemctl restart apache2
```


authentik.iespoveda.com/if/admin/#/core/providers
authentik
Proveedores
Proporcionar soporte para protocolos como SAML y OAuth a las aplicaciones asignadas.
Paneles
Aplicaciones
Aplicaciones
Proveedores
Puestos avanzados
Endpoint Devices
Buscar proveedores
Crear
Nombre
Aplicación
Tipo
Nuevo proveedor
Crear un nuevo proveedor.
1 Selecciona un tipo
Proveedor de Google Workspace
Sync users from authentik into Google Workspace.
Solo para Enterprise
Proveedor de LDAP
Allow applications to authenticate against authentik's users using LDAP.
Proveedor de Microsoft Entra
Sync users from authentik into Microsoft Entra.
Solo para Enterprise
Proveedor de OAuth2/OpenID
OAuth2 Provider for generic OAuth and OpenID Connect Applications.
Personal
Información personal
Seguridad
Notificaciones
Móvil y escritorio
Compartir
Apariencia y accesibilidad
Disponibilidad
Flujo
Privacidad
Administración
Vista general
Soporte
Ajustes básicos
Compartir
Seguridad
Tema
Inteligencia Artificial
Groupware
AppAPI
OpenID Connect
Permitir a los usuarios autenticarse por medio de proveedores OpenID connect.
Habilitar ID4me
Almacenar tokens de inicio de sesión
"Store login tokens" is needed if you are using other apps that want to use user_oidc's token exchange or simply get the login token
Proveedores Registrados
No hay proveedores registrados.



Crear nuevos usuarios ldap

Si el grupo ya estuviera generado:

```
# 1. Crear el usuario (si no existe) con su password
dn: cn=aortrei469,dc=iespoveda,dc=com
objectClass: inetOrgPerson
cn: arcadio
cn: aortrei469
sn: ortega
mail: aortrei469@g.educaand.es
userPassword: TU_PASSWORD_AQUI

# 2. Modificar el grupo existente para añadir al miembro
dn: ou=geducaand,dc=iespoveda,dc=com
changetype: modify
add: member
member: cn=aortrei469,dc=iespoveda,dc=com
```

Si tenemos que crear también el grupo:

```
-----------------------
```

```
# 1. Crear el usuario con su contraseña
dn: cn=aortrei469,dc=iespoveda,dc=com
objectClass: inetOrgPerson
cn: arcadio
cn: aortrei469
sn: ortega
mail: aortrei469@g.educaand.es
userPassword: TU_PASSWORD_AQUI

# 2. Crear el grupo geducaand e incluir al admin y al
usuario (DN correcto)
dn: ou=geducaand,dc=iespoveda,dc=com
objectClass: top
objectClass: groupOfNames
cn: geducaand
member: cn=admin,dc=iespoveda,dc=com
member: cn=aortrei469,dc=iespoveda,dc=com
ou: geducaand
```

Para cambiar el password:

```
ldappasswd -x -D "cn=admin,dc=iespoveda,dc=com" -W -S
"cn=aortrei469,dc=iespoveda,dc=com"
```

Automatizar Start/Stop máquinas virtuales

Tenemos las siguientes condiciones:

- Todas tienen ubuntu server instalado
- Todas tienen instalado y activo `qemu-guest-agent`
- Todas tienen una ip fija

Importante Hay que quitar el "autostart" de libvirt a cada una de esas máquinas:
Lo hacemos en cada una de las VMs de la lista

```
virsh autostart nombre_de_tu_vm --disable
```

O de forma automática

```
```

```
for vm in $(virsh list --all --name); do virsh autostart
"$vm" --disable; done
```

U obtener toda la lista mediante: `for vm in $(virsh list --all --name); do echo "$vm"; done`

virt... start y stop en orden nuestras MVs

Importante **Añadir en orden las máquinas virtuales afectadas**

```
cat /usr/local/bin/gestion-vms.sh
#!/bin/bash

# Configuración: Lista de VMs en orden de ARRANQUE
VMS=("database" "identity" "nextcloud" "fediverse"
"observability")
MAX_RETRIES=15  # Intentos (15 * 5s = 75s de espera máx
por VM)
LOG_FILE="/var/log/vms-ordenadas.log"

log() {
    echo "$(date '+%Y-%m-%d %H:%M:%S') - $1" | tee -a
"$LOG_FILE"
}

get_vm_ip() {
    local VM_NAME=$1
    # 1. Obtenemos datos del agente
    # 2. Filtramos por ipv4
    # 3. Excluimos interfaces: lo, docker, veth, br-
    # 4. Priorizamos nombres que empiecen por 'e' (enp,
eth)
    virsh domifaddr "$VM_NAME" --source agent 2>/dev/
null | \
    grep "ipv4" | \
    grep -vE "lo|docker|veth|br-" | \
    awk '/enp|eth|e/ {print $4; exit}' | cut -d/ -f1
}

start_vms() {
    log "--- Iniciando secuencia de arranque ---"
    for VM in "${VMS[@]}"; do
        log "Iniciando $VM..."
        virsh start "$VM" 2>/dev/null
```

```
        COUNT=0
        IP=""
        while [ -z "$IP" ] && [ $COUNT -lt $MAX_RETRIES
]; do
            sleep 5
            IP=$(get_vm_ip "$VM")

            if [ -n "$IP" ]; then
                log "IP detectada en interfaz física:
$IP. Verificando ping..."
                if ping -c 1 -W 1 "$IP" > /dev/null;
then
                    log "¡$VM está operativa y
respondiendo!"
                    break
                else
                    log "IP obtenida pero sin respuesta
a ping aún..."
                    IP=""
                fi
            fi
            COUNT=$((COUNT + 1))
            log "Esperando red en $VM... (intento
$COUNT/$MAX_RETRIES)"
        done
        [ -z "$IP" ] && log "ERROR: Timeout o no se
detectó IP válida en $VM."
    done
    log "=== SECUENCIA DE ARRANQUE COMPLETADA ==="
}

stop_vms() {
    log "--- Iniciando secuencia de apagado inverso ---"
    # Invertir el array de VMs
    REVERSE_VMS=($(echo "${VMS[@]}" | tr ' ' '\n' | tac
| tr '\n' ' '))

    for VM in "${REVERSE_VMS[@]}"; do
        log "Apagando $VM..."
        virsh shutdown "$VM" >/dev/null 2>&1

        # Esperar hasta que ya no aparezca en la lista
de VMs activas
        while virsh list --name | grep -qxw "$VM"; do
            sleep 2
        done
        log "$VM se ha detenido correctamente."
```

```
    done
    log "=== SECUENCIA DE PARADA COMPLETADA ==="
}

case "$1" in
    start)
        start_vms
        exit 0
        ;;
    stop)
        stop_vms
        exit 0
        ;;
    *)
        echo "Uso: $0 {start|stop}"
        exit 1
        ;;
esac
```

Ahora lo ponemos como servicio

```
cat /etc/systemd/system/vms-ordenadas.service
[Unit]
Description=Gestion de VMs en Orden con Delay y
Verificacion de IP
# Espera a que el demonio de virtualización esté activo
After=libvirtd.service
Requires=libvirtd.service

[Service]
Type=oneshot
# RemainAfterExit es clave: mantiene el servicio
"activo" así
# Systemd sabe que debe ejecutar el 'Stop' al apagar el
host.
RemainAfterExit=yes
ExecStart=/usr/local/bin/gestion-vms.sh start
ExecStop=/usr/local/bin/gestion-vms.sh stop
# Tiempo total para que todas las VMs cierren (ej: 10
min)
TimeoutStopSec=600

[Install]
WantedBy=multi-user.target
```

y damos de alta los logs del sistema y permitimos su ejecución:

```
sudo chmod +x /usr/local/bin/gestion-vms.sh
sudo touch /var/log/vms-ordenadas.log
sudo chmod 644 /var/log/vms-ordenadas.log
```

Comandos de Activación

Recargar la configuración:

```
sudo systemctl daemon-reload
```

Habilitar el servicio (Auto-inicio):

```
sudo systemctl enable vms-ordenadas.service
```

Probar manualmente (Opcional):

```
sudo systemctl start vms-ordenadas.service
```

dnsmasq.conf

Una vez que vamos teniendo las ethernets de cada máquina virtual:

```
admon@srv01:~$ cat /etc/dnsmasq.conf
# Configuration file for dnsmasq.
#
port=53
domain-needed
bogus-priv
strict-order
listen-address=127.0.0.1
interface=br0   # escucha solo en bridge de VMs
bind-interfaces
no-dhcp-interface=
domain=iespoveda.com
cache-size=1000
```

```
expand-hosts                # usa hostname + domain
#
host-record=fediverse.iespoveda.com,172.16.6.50
host-record=identity.iespoveda.com,172.16.6.10
host-record=nextcloud.iespoveda.com,172.16.6.40
host-record=database.iespoveda.com,172.16.6.20
host-record=observability,172.16.6.60
dhcp-
host=52:54:00:a4:53:a0,172.16.6.50,fediverse.iespoveda.com,infin
dhcp-
host=52:54:00:23:13:2e,172.16.6.10,identity.iespoveda.com,infini
dhcp-
host=52:54:00:95:0d:41,172.16.6.40,nextcloud.iespoveda.com,infin
dhcp-
host=52:54:00:1d:3f:6a,172.16.6.20,database.iespoveda.com,infini
dhcp-
host=52:54:00:13:31:62,172.16.6.60,observability.iespoveda.com,i
#
dhcp-range=172.16.6.20,172.16.6.100,12h  # rango DHCP
con lease de 12 horas
dhcp-option=3,172.16.6.1   # gateway para las VMs
dhcp-option=6,172.16.6.5   # DNS para las VMs (apunta a
identity)
# Entradas fijas para VMs conocidas (opcional)
#dhcp-host=identity,172.16.6.10
dhcp-host=core,172.16.6.30
#
cname=authentik.iespoveda.com,identity.iespoveda.com
cname=sso.iespoveda.com,identity.iespoveda.com
cname=auth.iespoveda.com,identity.iespoveda.com
#
cname=ldap.iespoveda.com,identity.iespoveda.com
#
cname=cloud.iespoveda.com,nextcloud.iespoveda.com
#
cname=mastodon.iespoveda.com,fediverse.iespoveda.com
#
cname=openproject.iespoveda.com,identity.iespoveda.com
```

MVs. Bases de datos

databases

```
admon@srv01:~$ sudo virsh domiflist database
 Interface   Type      Source   Model    MAC
-------------------------------------------------------------
 vnet0       bridge    br0      virtio
52:54:00:1d:3f:6a
```

Lo añadimos en nuestro servidor de nombres para que funciones vía dhcp4:

```
admon@srv01:~$ grep database /etc/dnsmasq.conf
host-record=database.iespoveda.com,172.16.6.20
dhcp-
host=52:54:00:1d:3f:6a,172.16.6.20,database.iespoveda.com,infini
```

Ahora entramos en el server y cambiamos su configuración: admon@srv01:~$ sudo virsh shutdown database Domain 'database' is being shutdown admon@srv01:~$ sudo guestmount -a /vm/libvirt/images/database.qcow2 -i /mnt admon@srv01:~$ sudo vi /mnt/etc/netplan/50-cloud-init.yamlCasi seguro que su device de red ethernet será: enp1s0

```
admon@srv01:~$ sudo cat /mnt/etc/netplan/50-cloud-
init.yaml
network:
  version: 2
  ethernets:
    enp1s0:
      dhcp4: true
```

```
admon@srv01:~$ sudo guestunmount /mnt
admon@srv01:~$ sudo virsh start database
```

```
Domain 'database' started

admon@srv01:~$ sudo virsh domiflist database
 Interface   Type     Source   Model    MAC
-------------------------------------------------------------
 vnet3       bridge   br0      virtio
52:54:00:1d:3f:6a

admon@srv01:~$ sudo virsh domifaddr database --source
agent
 Name        MAC address          Protocol     Address
-------------------------------------------------------------
 lo          00:00:00:00:00:00    ipv4
127.0.0.1/8
 -           -                    ipv6         ::1/128
 enp1s0      52:54:00:1d:3f:6a    ipv4
172.16.6.20/24
 -           -                    ipv6
fe80::5054:ff:fe1d:3f6a/64
 docker0     82:32:03:a9:90:25    ipv4
172.17.0.1/16
```

Preparamos los certificados de seguridad SSL Configuramos

```
root@srv01:~/certs# cat database.conf
[req]
default_bits       = 2048
distinguished_name = database.iespoveda.com
req_extensions     = v3_req
prompt             = no

[database.iespoveda.com]
C  = ES
ST = Andalucia
L  = Guadix
O  = IESPoveda
CN = database.iespoveda.com

[v3_req]
subjectAltName = @alt_names

[alt_names]
DNS.1 = database.iespoveda.com
root@srv01:~/certs# cat database.ext
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
```

```
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
DNS.1 = database.iespoveda.com
```

```
root@srv01:~/certs# openssl req -new -nodes -out
database.csr -keyout database.key -config database.conf
...
...
root@srv01:~/certs# openssl req -in database.csr -noout
-text | grep -A1 "Subject Alternative Name"
                X509v3 Subject Alternative Name:
                    DNS:database.iespoveda.com
root@srv01:~/certs# openssl x509 -req -in database.csr -
CA iespoveda.pem -CAkey iespoveda.key -CAcreateserial -
out database.crt -days 825 -sha256 -extfile database.ext
Certificate request self-signature ok
subject=C = ES, ST = Andalucia, L = Guadix, O =
IESPoveda, CN = database.iespoveda.com
```

Ahora entramos el la máquina virtual y nos traemos los certificados generados:

```
ubuntu@database:~$ ls -la /etc/ssl/database/
total 20
drwxr-xr-x 2 root  root  4096 mar 24 10:17 .
drwxr-xr-x 5 root  root  4096 mar 24 10:08 ..
-rw-r--r-- 1 mysql mysql 1874 mar 24 10:06 database.crt
-rw------- 1 mysql mysql 1708 mar 24 10:06 database.key
-rw-r--r-- 1 mysql mysql 2232 mar 24 10:17 iespoveda.pem
ubuntu@database:~$ ls -la /usr/local/share/ca-
certificates/iespoveda.crt
-rw-r--r-- 1 ubuntu ubuntu 2232 mar 24 10:06 /usr/local/
share/ca-certificates/iespoveda.crt
ubuntu@database:~$  sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does
not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

Ahora comenzamos la instalación

```
sudo apt update
sudo apt upgrade
sudo apt update && sudo apt install -y mariadb-server
postgresql openssl ufw
```

```
#!/bin/bash

set -e

SSL_DIR="/etc/ssl/database"

# CONFIGURAR MARIADB
# ----------------------------

echo "⚙  Configurando MariaDB con SSL..."
sudo sed -i 's/^bind-address.*/bind-address = 0.0.0.0/'
/etc/mysql/mariadb.conf.d/50-server.cnf

cat <<EOF | sudo tee -a /etc/mysql/mariadb.conf.d/50-
server.cnf

[mariadb]
ssl-ca=/etc/ssl/database/iespoveda.pem
ssl-cert=/etc/ssl/database/database.crt
ssl-key=/etc/ssl/database/database.key
require_secure_transport=ON
EOF

sudo systemctl restart mariadb

# Usuario que REQUIERE SSL

sudo mysql <<EOF
CREATE USER 'remote'@'%' IDENTIFIED BY
'DBMariaPgPass321' REQUIRE SSL;
GRANT ALL PRIVILEGES ON *.* TO 'remote'@'%' WITH GRANT
OPTION;
FLUSH PRIVILEGES;
EOF

# ----------------------------

# CONFIGURAR POSTGRESQL
```

```
# ----------------------------

echo "⚙  Configurando PostgreSQL con SSL..."

PG_VERSION=$(ls /etc/postgresql)

# Copiar certificados

sudo cp /etc/ssl/database/database.crt /etc/postgresql/
$PG_VERSION/main/
sudo cp /etc/ssl/database/database.key /etc/postgresql/
$PG_VERSION/main/
sudo cp /etc/ssl/database/iespoveda.pem /etc/postgresql/
$PG_VERSION/main/.
sudo chown postgres:postgres /etc/postgresql/
$PG_VERSION/main/database.*
sudo chown postgres:postgres /etc/postgresql/
$PG_VERSION/main/iespoveda.pem
sudo chmod 600 /etc/postgresql/$PG_VERSION/main/
database.key

# Activar SSL

sudo sed -i "s/^#listen_addresses =.*/listen_addresses =
'*'/" /etc/postgresql/$PG_VERSION/main/postgresql.conf
sudo sed -i "s/^#ssl = off/ssl = on/" /etc/postgresql/
$PG_VERSION/main/postgresql.conf

# Añadir acceso SSL obligatorio

echo "hostssl all all 0.0.0.0/0 md5" | sudo tee -a /etc/
postgresql/$PG_VERSION/main/pg_hba.conf

sudo systemctl restart postgresql

# Crear usuario

sudo -u postgres psql <<EOF
CREATE USER remote WITH PASSWORD 'DBMariaPgPass321';
ALTER USER remote WITH SUPERUSER;
CREATE DATABASE remotedb OWNER remote;
EOF

# ----------------------------

# FIREWALL
```

```
# ----------------------------

echo "Configurando firewall..."

sudo ufw allow 3306/tcp
sudo ufw allow 5432/tcp
#sudo ufw --force enable

# ----------------------------

#
```

Si no va las conexiones externas para mariadb:

```
sudo mysql -u root
DROP USER IF EXISTS 'remote'@'%';
DROP USER IF EXISTS 'remote'@'localhost';
DROP USER IF EXISTS 'remote'@'172.16.6.%';
CREATE USER 'remote'@'172.16.6.%' IDENTIFIED BY
'DBMariaPgPass321' REQUIRE SSL;
GRANT ALL PRIVILEGES ON *.* TO 'remote'@'172.16.6.%'
WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

y reiniciamos el servicio

```
sudo systemctl restart mariadb
```

Ahora lo pribamos en local:

```
ubuntu@database:~$ mysql -h 172.16.6.20 -u remote -p --
ssl-ca=/etc/ssl/database/iespoveda.pem
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or
\g.
Your MariaDB connection id is 34
...
```

Desde remoto:

```
admon@srv01:~$ mysql -h 172.16.6.20 -u remote -p --ssl-
```

```
ca=/usr/local/share/ca-certificates/iespoveda.crt Enter
password:
Welcome to the MariaDB monitor.  Commands end with ; or
\g.
Your MariaDB connection id is 35
...
...
```

Para dbeaver: añadir en la conexión, editar los datos, ssl pero no requerirlo ni verificarlo

Para conectarnos desde el server podemos usar:

```
ssh -i ~/.ssh/id_rsa -o StrictHostKeyChecking=no
ubuntu@172.16.6.20
```

MVs. observability

Grafana Loki

Página oficial: Grafana Loki
En dnsmasq.conf:

```
#
cname=loki.iespoveda.com,observability.iespoveda.com
cname=grafana.iespoveda.com,observability.iespoveda.com
cname=n8n.iespoveda.com,observability.iespoveda.com
```

Estoy en `/root`:

```
root@observability:~# cat promtail-config.yaml
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://loki:3100/loki/api/v1/push

scrape_configs:
- job_name: system
  static_configs:
  - targets:
      - localhost
    labels:
      job: varlogs
      __path__: /var/log/*.log
```

y

```
root@observability:~# cat docker-compose.yml
services:
  loki:
```

```
    image: grafana/loki:2.9.0
    container_name: loki
    restart: unless-stopped
    ports:
      - "3100:3100"
    volumes:
      - ./loki-config.yaml:/etc/loki/local-config.yaml
      - ./loki-data:/loki
    command: -config.file=/etc/loki/local-config.yaml

  grafana:
    image: grafana/grafana:latest
    container_name: grafana
    restart: unless-stopped
    ports:
      - "3000:3000"
    volumes:
      - ./grafana-data:/var/lib/grafana
    depends_on:
      - loki

  promtail:
    image: grafana/promtail:latest
    container_name: promtail
    restart: unless-stopped
    volumes:
      - /var/log:/var/log:ro
      - ./promtail-config.yaml:/etc/promtail/config.yml
    command: -config.file=/etc/promtail/config.yml
    depends_on:
      - loki

  n8n:
    image: docker.n8n.io/n8nio/n8n:latest
    container_name: n8n
    restart: unless-stopped
    ports:
      - "5678:5678"
    environment:
      - N8N_HOST=0.0.0.0
      - N8N_PORT=5678
      - N8N_PROTOCOL=http
      - NODE_ENV=production
      - WEBHOOK_URL=http://n8n.iespoveda.com:5678/
      - GENERIC_TIMEZONE=Europe/Madrid
      - EXECUTIONS_DATA_MAX_AGE=168
    volumes:
```

```
      - ./n8n-data:/home/node/.n8n
```

Ponemos en marcha:

```
docker compose up -d
docker ps -a
```

Verificamos: (puede tardar unos segundos)

```
root@observability:~# curl -G "http://localhost:3100/
ready"
ready
```

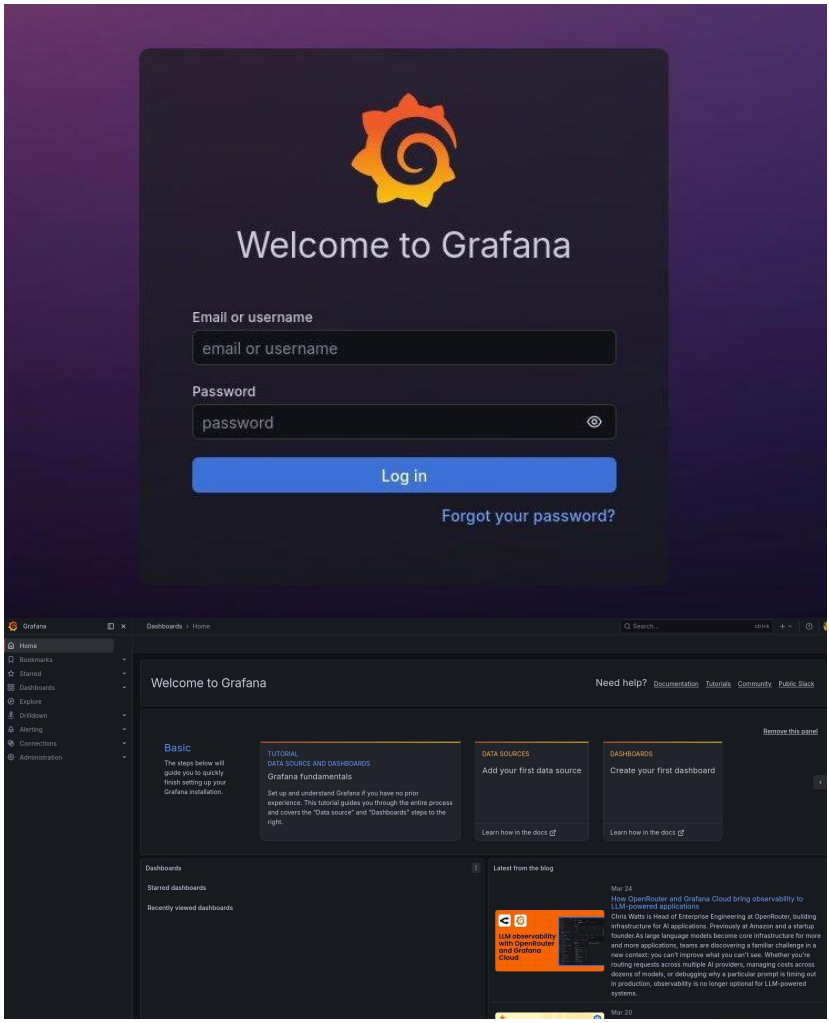
Al verificar n8n he tenido problemas.`Error: EACCES: permission denied, open '/home/node/.n8n/config'` Así que lo soluciono

```
chown -R 1000:1000 /root/n8n-data
chmod -R 775 /root/n8n-data
docker compose up -d n8n
```

y ahora verifico:

```
root@observability:~# curl -I http://localhost:5678
HTTP/1.1 200 OK
Accept-Ranges: bytes
Cache-Control: public, max-age=86400
Last-Modified: Fri, 27 Mar 2026 12:26:42 GMT
ETag: W/"3bd4-19d2f42d3fc"
Content-Type: text/html; charset=utf-8
Content-Length: 15316
Vary: Accept-Encoding
Date: Fri, 27 Mar 2026 12:29:00 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

Y ahora nos conectamos **Las credenciales son admin/admin**



n8n

Ahora configuramos n8n

```
root@observability:~# ss -tulnp | grep 5678
tcp   LISTEN 0      4096               0.0.0.0:5678
0.0.0.0:* users:(("docker-proxy",pid=9235,fd=7))
```

```
tcp   LISTEN 0      4096                [::]:5678
   [::]:* users:(("docker-proxy",pid=9242,fd=7))
```

Para tenerlo vía ssl, creamos un proxy inverso (usamos claves autofirmadas)

```
server {
    listen 80;
    server_name n8n.iespoveda.com grafana.iespoveda.com;
    return 301 https://$host$request_uri; # Redirigir
siempre a HTTPS
}

server {
    listen 443 ssl;
    server_name n8n.iespoveda.com;

    ssl_certificate     /etc/ssl/observability/
observability.crt;
    ssl_certificate_key /etc/ssl/observability/
observability.key;

    # Optimización SSL básica
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;

    location / {
        proxy_pass http://127.0.0.1:5678;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # Crucial para que n8n no se desconecte
(WebSockets)
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";

        proxy_buffering off;
        proxy_read_timeout 24h;
    }
}

server {
```

```
    listen 443 ssl;
    server_name grafana.iespoveda.com;

    ssl_certificate     /etc/ssl/observability/
observability.crt;
    ssl_certificate_key /etc/ssl/observability/
observability.key;

    location / {
        proxy_pass http://127.0.0.1:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Cambiamos el docker de n8n

```
 n8n:
    image: docker.n8n.io/n8nio/n8n:latest
    container_name: n8n
    restart: unless-stopped
    ports:
      - "5678:5678"
    environment:
      - N8N_HOST=0.0.0.0
      - N8N_PORT=5678
      - N8N_PROTOCOL=https
      - NODE_ENV=production
      - WEBHOOK_URL=https://n8n.iespoveda.com/
      - GENERIC_TIMEZONE=Europe/Madrid
      - EXECUTIONS_DATA_MAX_AGE=168
      - N8N_SECURE_COOKIE=false             # Evita
problemas de login con autofirmados
      - NODE_TLS_REJECT_UNAUTHORIZED=0      # Permite
que n8n hable con APIs HTTPS autofirmadas
```

y ejecutamos:

```
nginx -t
systemctl restart nginx
docker compose up -d n8n
```

Clientes que suministran datos

datasource en grafana

Primero damos de alta la fuente de datos en grafana

Ejemplo database.iespoveda.com

```
wget -q -O - https://apt.grafana.com/gpg.key | gpg --
dearmor > /etc/apt/keyrings/grafana.gpg
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg]
https://apt.grafana.com stable main" | tee /etc/apt/
sources.list.d/grafana.list
apt update
apt-get install promtail
ls /etc/promtail/
ls /etc/promtail/config.yml
```

```
cat config.yml
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
- url: http://observability.iespoveda.com:3100/loki/api/
v1/push

scrape_configs:
- job_name: system
  static_configs:
  - targets:
      - localhost
    labels:
      job: varlogs
      host: database
      __path__: /var/log/*log
```

y ahora lo ponemos en marcha:

```
systemctl status promtail.service
systemctl stop promtail.service
systemctl start promtail.service
```

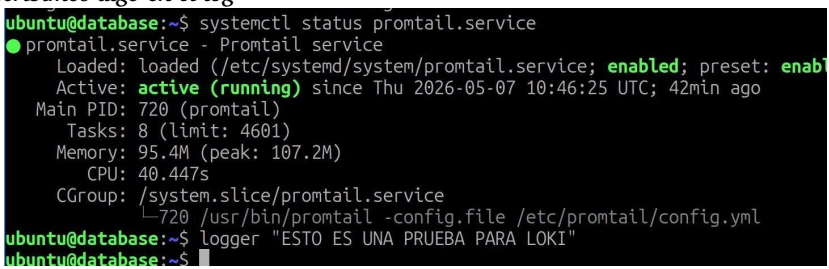
Podemos ver que todo va bien en:

```
journalctl -u promtail -f
```

Para forzar un mensaje puse:

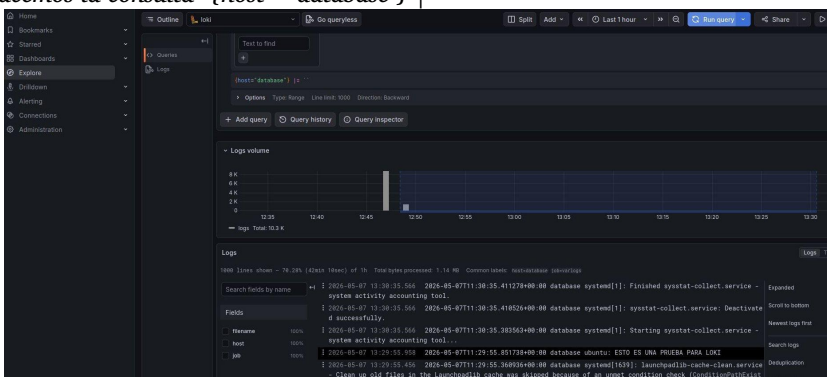
```
root@database:~#logger "ESTO ES UNA PRUEBA PARA LOKI"
```

Escribimos algo en el log



y vi que aparecía en grafana.

Hacemos la consulta "{host="database"} |= ``"



OpenProject

Para la instalación creamos la base de datos que va a necesitar el sevicio

```
ubuntu@database:~$ sudo -u postgres psql
psql (16.13 (Ubuntu 16.13-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# CREATE DATABASE openproject;
CREATE DATABASE
postgres=# CREATE USER openprj_user WITH PASSWORD
'openPrjSec321';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE openproject
TO openprj_user;
GRANT
postgres=# ALTER USER openprj_user CREATEDB;
ALTER ROLE
postgres=# \q
```

Necesitamos claves SSL. Las generamos como para los otros servicios en el servidor:

```
vi openproject.conf
vi openproject.ext
openssl req -new -nodes -out openproject.csr -keyout
openproject.key -config openproject.conf
openssl req -in openproject.csr -noout -text | grep -A1
"Subject Alternative Name"
openssl x509 -req -in openproject.csr -CA iespoveda.pem
-CAkey iespoveda.key -CAcreateserial -out
openproject.crt -days 825 -sha256 -extfile
openproject.ext
```

Lo damos de alta en nuestro DNS server:

```
admon@srv01:~$ grep openproject /etc/dnsmasq.conf
```

```
cname=openproject.iespoveda.com,identity.iespoveda.com
admon@srv01:~$ dig openproject.iespoveda.com

; <<>> DiG 9.18.39-0ubuntu0.24.04.2-Ubuntu <<>>
openproject.iespoveda.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
49385
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY:
0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;openproject.iespoveda.com.     IN      A

;; ANSWER SECTION:
openproject.iespoveda.com. 0    IN      CNAME
identity.iespoveda.com.
identity.iespoveda.com. 0       IN      A
172.16.6.10

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1) (UDP)
;; WHEN: Wed Mar 25 09:44:37 UTC 2026
;; MSG SIZE  rcvd: 106
```

Instalación del servicio

`file:///media/aor/TOSHIBA/POVEDA___2025_26/IAW/Modulo05_openproject.html`

Realizamos la instalación siguiendo nuestro documento y posteriormente realizamos los cambios

Cambios en nuestra MV

- Paramos apache2 porque lo vamos a poner con nginx. Hacemos

```
sudo systemctl stop apache2.service
sudo systemctl disable apache2.service
```

- Usamos la clave asimétrica autofirmada generada anteriormente

Va a compartir puerto: 443 con authentik, así:

```
ubuntu@identity:/etc/nginx/sites-enabled$ cat
openproject
# Proxy para OpenProject
server {
    listen 443 ssl;
    server_name openproject.iespoveda.com;

    ssl_certificate     /etc/ssl/openproject/
openproject.crt;
    ssl_certificate_key /etc/ssl/openproject/
openproject.key;
    ssl_protocols TLSv1.2 TLSv1.3;

    location / {
        proxy_pass http://127.0.0.1:6000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
ubuntu@identity:/etc/nginx/sites-enabled$ cat
redirecciona
# Opcional: redirigir HTTP a HTTPS
server {
    listen 80;
    server_name authentik.iespoveda.com
identity.iespoveda.com;

    return 301 https://$host$request_uri;
}
```

Verificamos que todos los servicios estén andando

```
ubuntu@identity:/opt/openproject$  systemctl status
openproject
openproject.service           openproject-web.service
   openproject-worker.service
openproject-web-1.service     openproject-
```

```
worker-1.service
ubuntu@identity:/opt/openproject$  systemctl status
openproject-web
● openproject-web.service
     Loaded: loaded (/etc/systemd/system/openproject-
web.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-25
10:14:29 UTC; 48s ago
   Main PID: 15601 (sleep)
      Tasks: 1 (limit: 3391)
     Memory: 196.0K (peak: 760.0K)
        CPU: 4ms
     CGroup: /system.slice/openproject-web.service
             15601 /bin/sleep infinity
ubuntu@identity:/opt/openproject$ systemctl status
openproject-web
openproject-web-1.service  openproject-web.service
ubuntu@identity:/opt/openproject$ systemctl status
openproject-web.service
● openproject-web.service
     Loaded: loaded (/etc/systemd/system/openproject-
web.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-25
10:14:29 UTC; 1min 4s ago
   Main PID: 15601 (sleep)
      Tasks: 1 (limit: 3391)
     Memory: 196.0K (peak: 760.0K)
        CPU: 4ms
     CGroup: /system.slice/openproject-web.service
             15601 /bin/sleep infinity
ubuntu@identity:/opt/openproject$ systemctl status
openproject-worker
openproject-worker-1.service  openproject-worker.service

ubuntu@identity:/opt/openproject$ systemctl status
openproject-worker
● openproject-worker.service
     Loaded: loaded (/etc/systemd/system/openproject-
worker.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-25
10:14:29 UTC; 1min 16s ago
   Main PID: 15603 (sleep)
      Tasks: 1 (limit: 3391)
     Memory: 196.0K (peak: 1.2M)
        CPU: 6ms
     CGroup: /system.slice/openproject-worker.service
             15603 /bin/sleep infinity
```

```
ubuntu@identity:/opt/openproject$ systemctl status
openproject-worker-1.service
● openproject-worker-1.service
     Loaded: loaded (/etc/systemd/system/openproject-
worker-1.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-25
10:14:29 UTC; 1min 19s ago
   Main PID: 15604 (worker)
      Tasks: 12 (limit: 3391)
     Memory: 612.3M (peak: 612.5M)
        CPU: 49.007s
     CGroup: /system.slice/openproject-worker-1.service
             15604 /bin/bash -e ./packaging/scripts/
worker
             15668 "/opt/openproject/vendor/bundle/
ruby/3.4.0/bin/good_job start"
ubuntu@identity:/opt/openproject$ sudo ss -tulipn | grep
6000
tcp   LISTEN 0      1024            127.0.0.1:6000
0.0.0.0:*    users:(("ruby",pid=16023,fd=7),
("ruby",pid=16010,fd=7),("ruby",pid=15658,fd=7))
```

Para no tener problemas con el fqdn he hecho:

```
sudo openproject config:set
OPENPROJECT_HOST__NAME="openproject.iespoveda.com"
sudo openproject config:set OPENPROJECT_HTTPS="true"
sudo openproject config:set PORT="6001"
sudo openproject config:set HOST="127.0.0.1"
sudo openproject config:set
OPENPROJECT_ADDITIONAL__HOST__NAMES='["openproject.iespoveda.com",
"127.0.0.1", "localhost"]'
sudo openproject configure
```

Y ahora reiniciamos servicios:

```
sudo openproject restart
ubuntu@identity:/opt/openproject$ sudo systemctl restart
nginx.service
ubuntu@identity:/opt/openproject$ sudo systemctl status
nginx.service
● nginx.service - A high performance web server and a
reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/
nginx.service; enabled; preset: enabled)
```

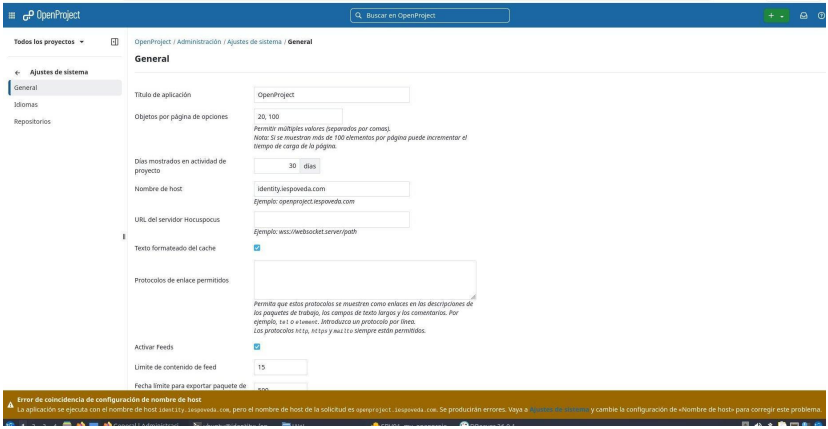
```
     Active: active (running) since Wed 2026-03-25
10:16:18 UTC; 1s ago
       Docs: man:nginx(8)
    Process: 16285 ExecStartPre=/usr/sbin/nginx -t -q -g
daemon on; master_process on; (code=exited, sta>
    Process: 16287 ExecStart=/usr/sbin/nginx -g daemon
on; master_process on; (code=exited, status=0/SUC>
   Main PID: 16289 (nginx)
      Tasks: 3 (limit: 3391)
     Memory: 3.7M (peak: 3.8M)
        CPU: 56ms
     CGroup: /system.slice/nginx.service
             16289 "nginx: master process /usr/sbin/
nginx -g daemon on; master_process on;"
             16290 "nginx: worker process"
             16291 "nginx: worker process"

mar 25 10:16:18 identity systemd[1]: Starting
nginx.service - A high performance web server and a
revers>
mar 25 10:16:18 identity systemd[1]: Started
nginx.service - A high performance web server and a
reverse>
```

Nos conectamos:

Creedenciales: Usuario: admin Contraseña: admin

> *Nota importante: En cuanto entres por primera vez, el sistema te obligará a cambiar esa contraseña por una segura. Si por algún motivo no funcionara:*
>
> ```
> sudo openproject run admin_user_reset
> ```

Al cambiar la contraseña tengo: "Error de coincidencia de configuración de nombre de host La aplicación se ejecuta con el nombre de host identity.iespoveda.com, pero el nombre de host de la solicitud es openproject.iespoveda.com. Se producirán errores. Vaya a Ajustes de sistema y cambie la configuración de «Nombre de host» para corregir este problema. " por lo que tengo que igualar los dos nombres:




```
sudo vi /etc/openproject/conf.d/other
...
...
export RECOMPILE_RAILS_ASSETS="""
export OPENPROJECT_HOST__NAME="identity.iespoveda.com"
export OPENPROJECT_HTTPS="true"
export OPENPROJECT_HSTS="true"
...
```

Cambiamos el hostname anterior y hacemos

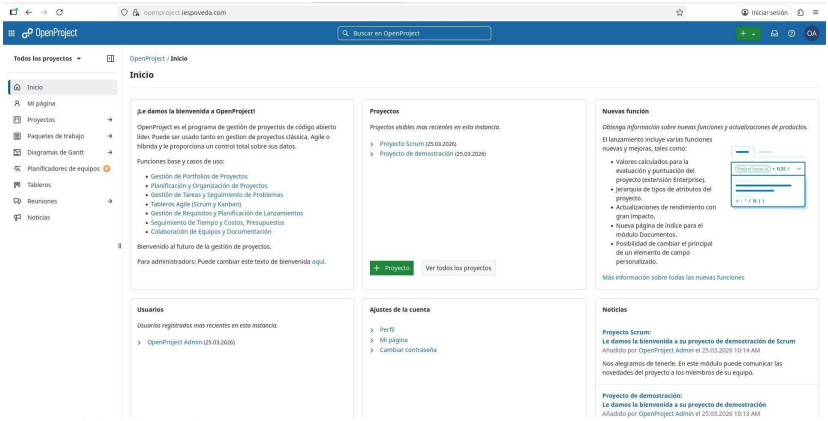
```
sudo openproject restart
```

Ver que tarda un poco en iniciar

```
ubuntu@identity:/opt/openproject$ sudo systemctl restart
nginx.service ubuntu@identity:/opt/openproject$ curl -I
-H "Host: openproject.iespoveda.com"
http://127.0.0.1:6001curl: (7) Failed to connect to
127.0.0.1 port 6001 after 0 ms: Couldn't connect to
server
ubuntu@identity:/opt/openproject$ curl -I -H "Host:
openproject.iespoveda.com" http://127.0.0.1:6001
curl: (7) Failed to connect to 127.0.0.1 port 6001 after
0 ms: Couldn't connect to server
ubuntu@identity:/opt/openproject$ curl -I -H "Host:
openproject.iespoveda.com" http://127.0.0.1:6001
curl: (7) Failed to connect to 127.0.0.1 port 6001 after
0 ms: Couldn't connect to server
```

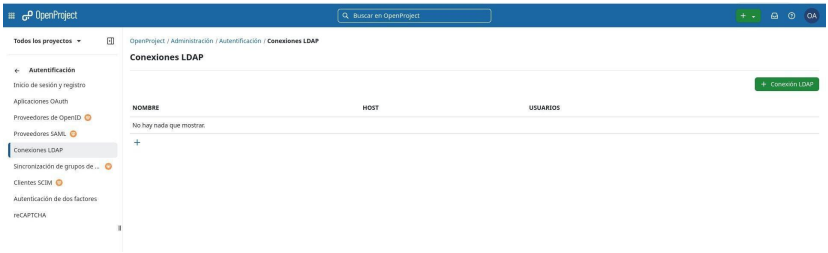
```
ubuntu@identity:/opt/openproject$
ubuntu@identity:/opt/openproject$ curl -I -H "Host:
openproject.iespoveda.com" http://127.0.0.1:6001
HTTP/1.1 301 Moved Permanently
Content-Type: text/html; charset=utf-8
Location: https://openproject.iespoveda.com/
X-Frame-Options: SAMEORIGIN
Content-Length: 0
```

y ya lo tenemos:



Ldap y Openproject

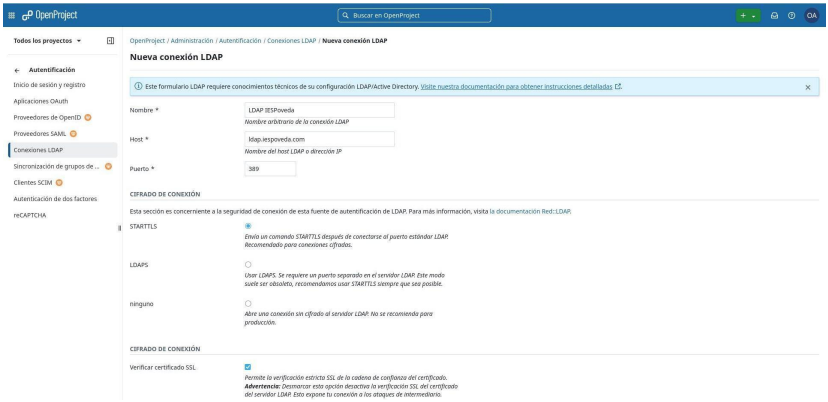
Usuarios posix

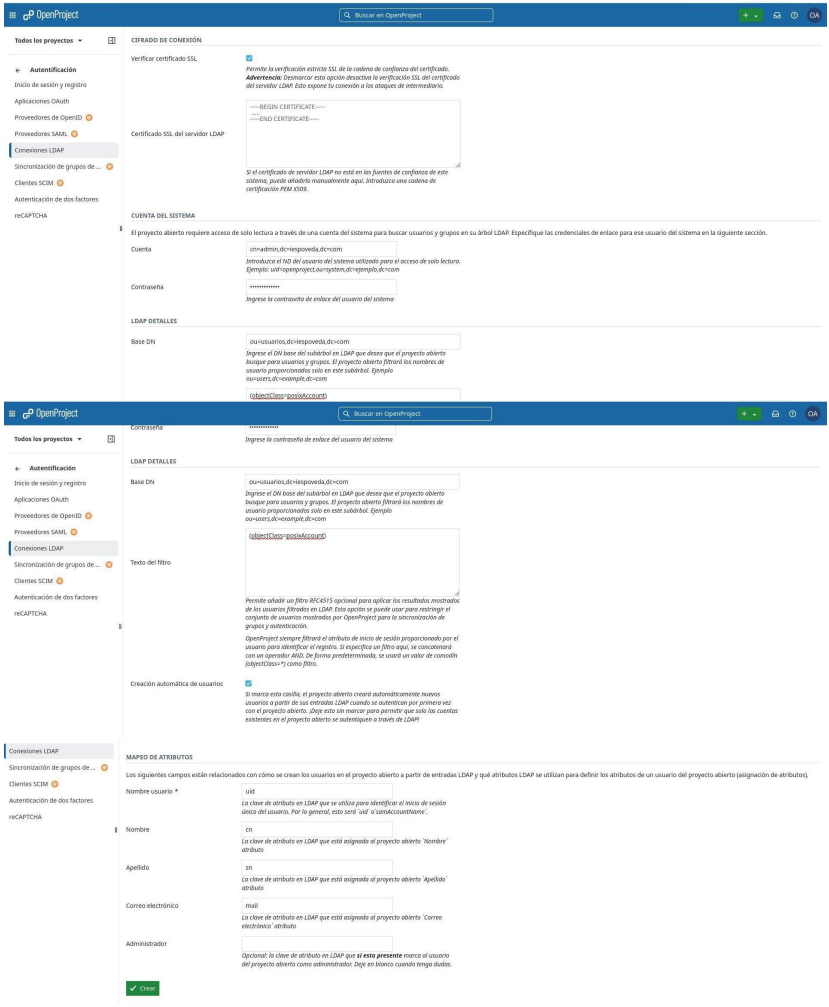


Valores que suministramos (Resumen de la tabla de mapeo)

- Nombre usuario uid
- Nombre cn
- Apellido sn (o cn si no tienes sn)
- Correo electrónico mail
- Administrador (vacío)

Porqué dejamos el administrador vacío:
"A menos que tengamos un atributo específico en LDAP (como isProjectAdmin: true), es mejor gestionar quién es administrador directamente desde el panel de control de OpenProject una vez que el usuario haya entrado por primera vez."





Importante: mail: o lo tenemos o no nos dejará entrar
Es importante ver si tenemos correo electrónico en los usuarios
Ejemplo para uno de mis usuarios:

```
ldapsearch -x -D "cn=admin,dc=iespoveda,dc=com" -W -b
"uid=arcadio,ou=usuarios,dc=iespoveda,dc=com" mail sn
```

Se lo añadimos antes de seguir:

```
dn: uid=arcadio,ou=usuarios,dc=iespoveda,dc=com
```

```
changetype: modify
add: mail
mail: arcadio@iespoveda.com
-
add: sn
sn: ApellidoDeArcadio
```

Aplicamos el cambio:

```
ubuntu@identity:~/ldifs_ldap$ ldapmodify -x -D
"cn=admin,dc=iespoveda,dc=com" -W -f
arcadio_modifica_mail.ldif
Enter LDAP Password:
modifying entry
"uid=arcadio,ou=usuarios,dc=iespoveda,dc=com"
```

Ahora probamos la conexión:

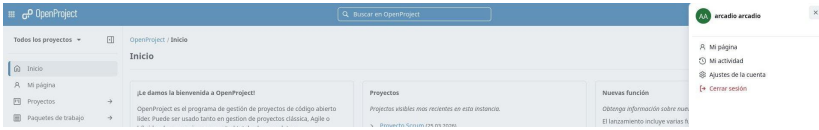
Ahora intentamos:

El primer inicio de sesión

Como marcamos la casilla de "Creación automática de usuarios", cuando el usuario, por ejemplo arcadio, intente entrar por primera vez:

- OpenProject verificará su contraseña contra el LDAP.
- Si es correcta, OpenProject creará una cuenta local para él automáticamente.
- El usuario entrará al sistema, pero estará "vacío" (no verá proyectos todavía).

El usuario entra por vez primera



Notas a tener en cuenta

- Sincronización: Si cambias el nombre o el correo en el LDAP, OpenProject actualizará esos datos la próxima vez que el usuario inicie sesión.
- Grupos de OpenProject: Aunque el usuario ya esté en un "grupo POSIX" en el servidor, OpenProject no lo mete automáticamente en sus propios grupos internos. Si quieres gestionar permisos de forma masiva, tendrás que crear los Grupos dentro de OpenProject (Administración > Grupos) y añadir a los usuarios LDAP allí.
- Contraseñas: Si un usuario olvida su clave, debe cambiarla en el servidor LDAP (el Ubuntu identity), no en OpenProject.

n8n. MV: observability

n8n

Cargar el json en n8n: El método rápido (Copiar y Pegar)

Es la forma más común y directa:

- Copiamos el código JSON.
- Abrimos el editor de flujos en n8n.
- Hacemos clic en cualquier parte vacía del lienzo (el fondo cuadriculado).
- Presionamos Ctrl + V (en Windows/Linux) o Cmd + V (en Mac).
- Ahora los nodos aparecen automáticamente en pantalla.

Probamos la comunicación y nos da error
No estaba cargando loki los parámetros. Pero antes para ver los errores probamos que la comunicación vaya bien:

- En el mismo server de n8n ponemos:

```
ubuntu@observability:~$ curl -G "http://
loki.iespoveda.com:3100/loki/api/v1/query" \
  --data-urlencode 'query={host="database",
job="varlogs"}'
{"status":"success","data":
{"resultType":"streams","result":[],"stats":
{"summary":
{"bytesProcessedPerSecond":0,"linesProcessedPerSecond
{"store":
{"totalChunksRef":0,"totalChunksDownloaded":0,"chunks
{"headChunkBytes":0,"headChunkLines":0,"decompressed
{"totalReached":1,"totalChunksMatched":0,"totalBatche
{"totalChunksRef":0,"totalChunksDownloaded":0,"chunks
{"headChunkBytes":0,"headChunkLines":0,"decompressed
{"chunk":
{"entriesFound":0,"entriesRequested":0,"entriesStored
```

```
{"entriesFound":0,"entriesRequested":0,"entriesStored":0,"k
{"entriesFound":0,"entriesRequested":0,"entriesStored":0,"k
{"entriesFound":0,"entriesRequested":0,"entriesStored":0,"k
```

- Y en el browser desde que nos conectamos a n8n abrimos una nueva pestaña y ponemos: `http://172.16.6.60:3100/loki/api/v1/query?query={host=%22database%22}` con la ip del loki server. Deber de devolvernos "success"

Ahora configuramos nuestro elemento en n8n. Une vez que vemos que todo funciona bien

Parámetros:

- start: He puesto una expresión en JavaScript: {{ Math.floor((Date.now() - 300000) * 1000000) }}. Esto busca logs de los últimos 5 minutos (Loki usa nanosegundos, por eso multiplicamos por 1.000.000.)
- LogQL: La query está configurada como {host = "database", job = "varlogs"}

Seguridad. Importante

"pequeño secreto" de la arquitectura de Grafana Labs: **Loki, por defecto, no tiene autenticación.**

Está diseñado bajo la filosofía de "seguridad por capas". Se asume que Loki vive en una red interna privada y que solo Grafana (o en tu caso, n8n) puede llegar a él.
Si alguien conoce la IP y el puerto 3100, puede leer todos tus logs.

Para solucionarlo:

1. El método profesional: Reverse Proxy (Nginx)

La forma estándar de proteger Loki es poner un **Nginx** o un **Apache** delante.

- El proxy pide un **Basic Auth** (usuario y contraseña).
- Si las credenciales son correctas, el proxy pasa la petición a Loki.
- En **n8n**, tendrías que cambiar la autenticación del nodo HTTP de "None" a "Basic Auth" y poner tus credenciales.

2. El método "Docker" (Si usas contenedores)

Si n8n y Loki están en la misma máquina, lo ideal es:

- **No abrir el puerto 3100 al exterior**: En tu `docker-compose.yml`, en lugar de:

  ```
  ports:
    - "3100:3100"
  ```

  Usa simplemente:

  ```
  expose:
    - "3100"
  ```

- De esta forma, Loki solo es visible para otros contenedores en la misma red (como n8n), pero si alguien intenta entrar desde fuera por la IP de la máquina, el puerto estará cerrado.

3. Modo Multi-tenant (Cabecera `X-Scope-OrgID`)

Loki permite activar un modo llamado "multi-tenant". No es una contraseña propiamente dicha, pero te obliga a enviar una cabecera HTTP llamada `X-Scope-OrgID`. Si no la envías, Loki rechaza la conexión. Es una capa mínima de seguridad para separar datos de distintos clientes.

4. Firewall simple

Si no quieres complicarte con Nginx, asegúrate de que el firewall de tu servidor (`ufw` o `iptables`) solo permita conexiones al puerto 3100 desde:

- La IP de tu servidor de Grafana.

- La IP de tu servidor de n8n.
- **Bloquea todo lo demás.**

Trabajar con n8n

Nodos (o conectores) esenciales de N8N, organizados en categorías lógicas

1. Triggers (Desencadenadores)

Representan el inicio de la ejecución de una automatización.

- **Manual Trigger:** Es el más simple; inicia el flujo cuando el usuario presiona un botón. Se utiliza principalmente en el diseño para probar piezas del workflow o para generar reportes bajo demanda.
- **Schedule Trigger:** Funciona como un despertador, ejecutando el flujo en momentos específicos definidos por el usuario (segundos, minutos, horas, etc.). Es ideal para copias de seguridad o sincronización periódica de datos.
- **Chat Trigger:** Inicia el flujo cuando un usuario envía un mensaje desde una interfaz de chat. Es la base para construir chatbots con IA.
- **Trigger de aplicación (ej. WhatsApp):** Se activa ante eventos externos en aplicaciones como WhatsApp, Gmail o Drive. Representan "callbacks" donde la app externa notifica a N8N que algo ocurrió.
    - **1. El Trío de Productividad (Google & Microsoft)**

      Son los reyes absolutos porque casi cualquier flujo de negocio empieza o termina en una hoja de cálculo o un correo.

- **Gmail / Outlook (Message Received):** Se activa cuando llega un correo. Es la base para sistemas de "Auto-Reply" con IA o para extraer facturas de archivos adjuntos automáticamente.
- **Google Sheets (Row Added/Updated):** Ideal para cuando alguien rellena manualmente una fila o cuando otra herramienta (como un formulario simple) inserta datos.
- **Google Drive (File Created):** Muy usado para detectar cuando se sube un nuevo documento (PDF, imagen) y procesarlo inmediatamente con OCR o IA.

○ **2. Comunicación y Notificaciones**

Fundamentales para la interacción en tiempo real con equipos o clientes.

- **Slack / Discord (Message Received):** No solo para notificaciones, sino para crear "comandos" internos. Por ejemplo: escribes `/analizar [url]` en Slack y el trigger dispara un flujo que resume esa web.
- **Telegram (Message):** Es el favorito para flujos personales o bots de bajo coste, ya que su API es muy abierta y fácil de configurar.
- **WhatsApp (via Twilio o Vonage):** Aunque mencionaste WhatsApp, los triggers a través de proveedores oficiales son los más robustos para empresas que reciben miles de mensajes de clientes.

○ **3. CRM y Ventas**

El motor de los ingresos. Estos triggers permiten que el equipo de ventas no tenga que mover datos a mano.

- **HubSpot / Salesforce (Object Created/ Updated):** Se activa cuando entra un nuevo "Lead" o cuando un trato cambia de etapa. Permite disparar flujos de bienvenida o

asignación de comerciales.
    - **Stripe (Payment Succeeded):** Crucial para el e-commerce. Cuando alguien paga, este trigger activa el envío de la factura, el acceso al curso o el mensaje de agradecimiento.

- **4. Gestión de Proyectos y Bases de Datos No-Code**

  Donde vive la "verdad" de la empresa.

    - **Notion (Page Added to Database):** Se ha vuelto masivo. Cada vez que añades una tarea o un cliente en Notion, n8n puede enriquecer esa página con datos de la web o avisar al equipo.
    - **Airtable (Record Created/Updated):** Similar a Sheets, pero mucho más potente para gestionar estados (ej. "Pasar de 'Borrador' a 'Publicado'").
    - **Typeform / Jotform:** El inicio de casi todo embudo de marketing. Captura los datos del formulario en el segundo en que el usuario hace clic en "Enviar".

2. Datos y Almacenamiento

Nodos para organizar y guardar la información, evitando que los datos desaparezcan tras la ejecución.

- **Edit Fields:** Permite crear, modificar y organizar datos que pasan al siguiente nodo. Representa la asignación de valores a variables y es útil para limpiar o transformar datos. No tiene persistencia fuera de la ejecución.
- **Google Sheets:** Utilizado para leer, escribir, actualizar y eliminar datos en hojas de cálculo. Es una base de datos fácil para empezar, aunque se ralentiza con más de 10,000 filas.
- **Data Table:** Es una base de datos interna de N8N que persiste entre ejecuciones. Permite configurar campos con tipos de datos específicos sin depender de servicios externos.

3. Control de Flujo

- **Split Out:** Descompone una lista compleja en elementos individuales para procesarlos uno a uno. Por ejemplo, separa los productos de un pedido para tratarlos por separado.
- **Aggregate:** Es el opuesto a Split Out; agrupa elementos individuales en una sola lista organizada. Útil para consolidar ventas en un solo reporte.
- **Merge:** Combina datos de múltiples fuentes o flujos basándose en criterios específicos, similar a un "join" en SQL. Se usa para enriquecer datos de clientes uniendo información de distintas fuentes.

4. Toma de Decisiones

- **If:** Divide el flujo en dos caminos (verdadero o falso) según se cumpla una condición simple. Ejemplo: enviar un mail en castellano si el país es España, o en inglés si no lo es.
- **Switch:** Dirige el flujo por múltiples caminos (más de dos) según diversas condiciones. Funciona como una "rotonda" para categorizar datos en varios grupos.
- **Loop Over Items:** Procesa datos en grupos más pequeños o lotes de manera repetitiva. Es crítico para evitar saturar servidores o cumplir con límites de APIs.
- **Filter:** Actúa como un colador, dejando pasar solo los elementos que cumplen ciertos criterios y descartando el resto. A diferencia del *If*, solo tiene una salida de datos útiles.
- **Limit:** Restringe la cantidad máxima de elementos que pasan al siguiente nodo. Es útil para mostrar un "top 5" o para hacer pruebas rápidas con un solo ítem durante el desarrollo.

5. Integración

- **Webhook:** Un "timbre" que espera a ser llamado por una URL externa para iniciar el flujo. Es la puerta de entrada para notificaciones de e-commerce o formularios web.
- **HTTP Request:** Permite a N8N llamar a otras APIs o servicios web para pedir o enviar datos. Es el nodo más versátil para conectar con cualquier herramienta que tenga API.
- **Respond to Webhook:** Controla qué respuesta se envía de

vuelta a quien llamó al Webhook, completando el ciclo de comunicación bidireccional.

6. Inteligencia Artificial (IA)

- **Basic LLM Chain (Chain):** Procesamiento lineal donde se da una entrada a la IA y se recibe una salida. No tiene memoria ni herramientas, por lo que es ideal para procesar textos en masa.
- **AI Agent:** Un sistema autónomo que puede tomar decisiones, usar herramientas (búsqueda web, calculadoras) y disponer de memoria para resolver tareas complejas.

El Nodo "Especial" (Bonus)

- **Code Node:** Permite escribir código en **JavaScript o Python** directamente en el flujo. Es el puente entre el no-code y la programación tradicional, ofreciendo flexibilidad total para cálculos complejos o transformaciones de datos avanzadas que los nodos visuales no permiten.

7. Gestión de IA Avanzada (RAG)

Si vas a usar el **AI Agent**, necesitas alimentar su "conocimiento". Estos nodos son el estándar para crear sistemas que consulten tus propios documentos:

- **Vector Store (ej. Pinecone, Qdrant, Supabase):** Es el "cerebro" donde la IA busca información relevante antes de responder. Sin esto, la IA solo sabe lo que aprendió en su entrenamiento original.
- **Embeddings (OpenAI/Mistral):** El traductor que convierte texto humano en números (vectores) para que la base de datos vectorial pueda entender similitudes.
- **Document Loader (PDF, Web, Text):** Crucial para importar manuales, PDFs o webs enteras al flujo de IA.

---

8. Robustez y Mantenimiento

Un flujo que funciona hoy pero falla mañana sin avisar no es profesional. Estos nodos gestionan la salud de tu automatización:

- **Error Trigger:** Este es un nodo "invisible". Se activa automáticamente cuando **cualquier otro flujo falla**. Es vital para que n8n te envíe un mensaje (por Slack, Telegram o Email) avisándote de que algo se rompió.
- **Wait:** A veces las APIs externas tardan en procesar algo. Este nodo permite pausar el flujo unos segundos o minutos para evitar errores de "datos no encontrados".
- **Execute Workflow:** Permite llamar a un flujo dentro de otro flujo. Es la clave para la **modularidad**: en lugar de tener un flujo gigante e inmanejable, tienes piezas pequeñas que se llaman entre sí.

---

9. Utilidades de Datos y Formato

A menudo los datos no vienen "limpios". Estos nodos te ahorran horas de peleas con el formato:

- **HTML Extractor:** Si haces *scraping* o recibes correos en HTML, este nodo te permite extraer solo el texto o los enlaces que te interesan de forma visual.
- **Binary Data (Read/Write Binary File):** Esencial para trabajar con **imágenes, PDFs o archivos Excel**. Si necesitas descargar una factura de un correo y subirla a Drive, este nodo es el que maneja el "archivo" físico.
- **Crypto:** Fundamental para la seguridad. Sirve para crear hashes, firmas digitales o cifrar datos sensibles antes de guardarlos.

---

10. Comunicación Directa

A parte de los vistos de comunicación como WhatsApp, hay dos que son el estándar en operaciones internas:

- **Slack / Discord:** No solo para chatear, sino para crear **canales de log**. Cada vez que una venta ocurre o un cliente se registra, n8n envía un mensaje estructurado aquí.
- **Gmail / Outlook:** Siguen siendo los reyes de la automatización de oficina. Leer adjuntos automáticamente o enviar respuestas personalizadas basadas en IA.

---

wordpress

Instalación

Instalar dependencias

```
sudo apt update && sudo apt upgrade -y
sudo apt install apache2 mysql-server php php-mysql
libapache2-mod-php php-xml php-mbstring unzip curl -y
```

Crear base de datos y dar de alta usuario `sudo mysql -u root -p`

```
CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
CREATE USER 'wpuser'@'localhost' IDENTIFIED BY
'clave_segura';
GRANT ALL PRIVILEGES ON wordpress.* TO
'wpuser'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

Verificar que todo está hecho:

```
aor@wordpress:/tmp$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or
\g.
Your MySQL connection id is 10
Server version: 8.0.43-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation
and/or its
affiliates. Other names may be trademarks of their
respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the
```

```
current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| wordpress          |
+--------------------+
5 rows in set (0.01 sec)

mysql> SHOW GRANTS FOR 'wpuser'@'localhost';
+-----------------------------------------------------------------
+
| Grants for wpuser@localhost
       |
+-----------------------------------------------------------------
+
| GRANT USAGE ON *.* TO `wpuser`@`localhost`
       |
| GRANT ALL PRIVILEGES ON `wordpress`.* TO
`wpuser`@`localhost` |
+-----------------------------------------------------------------
+
2 rows in set (0.00 sec)

mysql> exit;
Bye
aor@wordpress:/tmp$ mysql -u wpuser -p wordpress
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or
\g.
Your MySQL connection id is 11
Server version: 8.0.43-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation
and/or its
affiliates. Other names may be trademarks of their
respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the
```

```
current input statement.

mysql> exit;
Bye
```

Descargamos wordpress

```
cd /tmp
curl -O https://wordpress.org/latest.tar.gz
tar xzvf latest.tar.gz
sudo mv wordpress /var/www/html/
sudo chown -R www-data:www-data /var/www/html/wordpress
sudo chmod -R 755 /var/www/html/wordpress
```

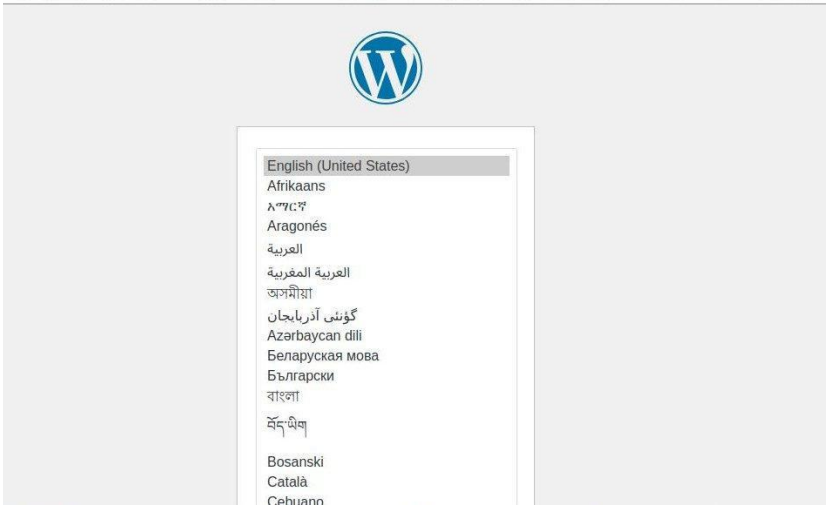
Configuramos apache: `sudo vi /etc/apache2/sites-available/wordpress.conf`

```
<VirtualHost *:80>
    ServerName wordpress.local
    DocumentRoot /var/www/wordpress

    <Directory /var/www/wordpress>
        AllowOverride All
    </Directory>
</VirtualHost>
```

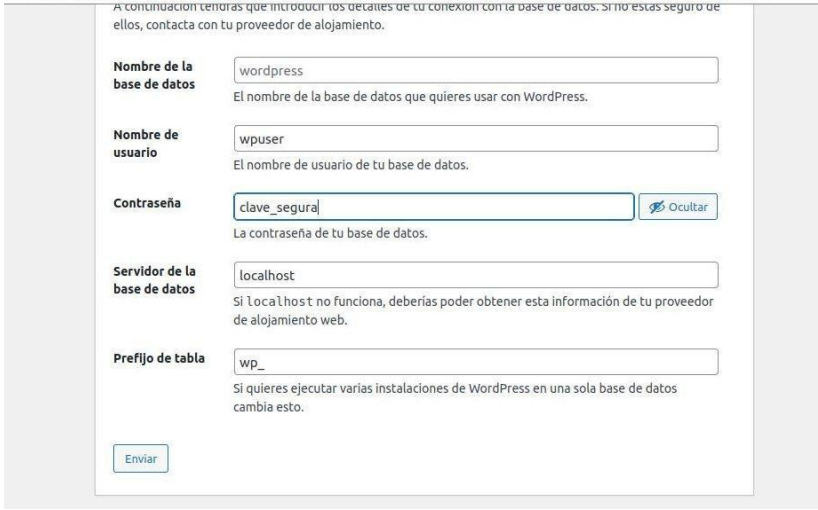
```
sudo a2dissite 000-default.conf
sudo a2ensite wordpress.conf
sudo a2enmod rewrite
sudo systemctl reload apache2
```

```
aor@wordpress:/tmp$ ip a
...
...
    inet 192.168.1.35/24 metric 100 brd 192.168.1.255
scope global dynamic enp0s3
...
...
```

Nos conectamos a `http://192.168.1.35/` *Pantalla de inicio wordpress*



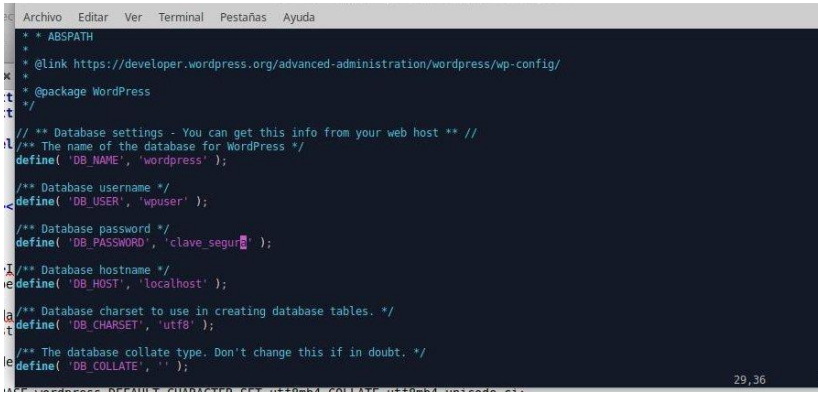
Si tenemos problemas a la hora de configurarlo podemos hacerlo también a mano Pinchando aquí *Pantalla de configuración*



Si no funciona probamos con 127.0.0.1 en lugar de localhost o lo hacemos a mano:

```
cd /var/www/html/wordpress
sudo cp wp-config-sample.php wp-config.php
sudo chown www-data:www-data wp-config.php
sudo vi wp-config.php
```

Instalación manual



Reiniciamos la página *El sistema verifica que ya estaba configurado*

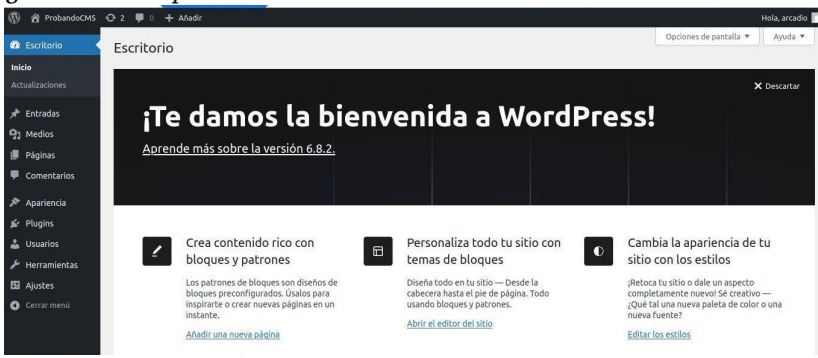
Y ahora llegamos a: *Creación del site*

Datos para la creación del site

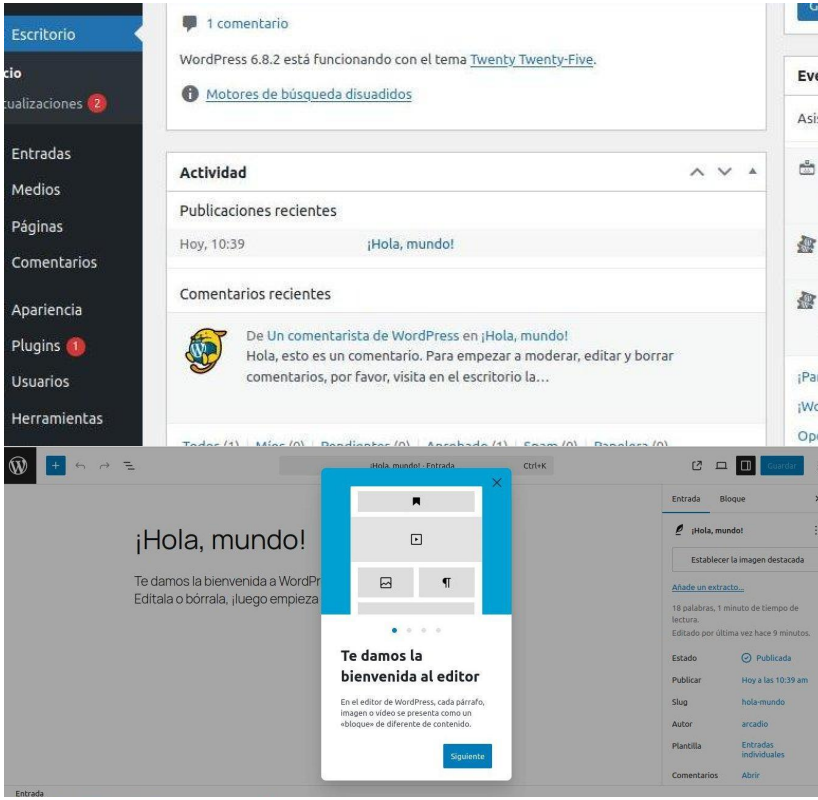
clave de arcadio: 5XJg46z@Xl@UjpDzoE
Le damos a instalar wordpress:

Pantalla de acceso a wordpress

Página inicio wordpress



Para probar por ejemplo vamos a cambiar la entrada de "Hola mundo":



Y podemos ver los resultados y como gestionarlo: *Gestión del contenido*
Página de inicio

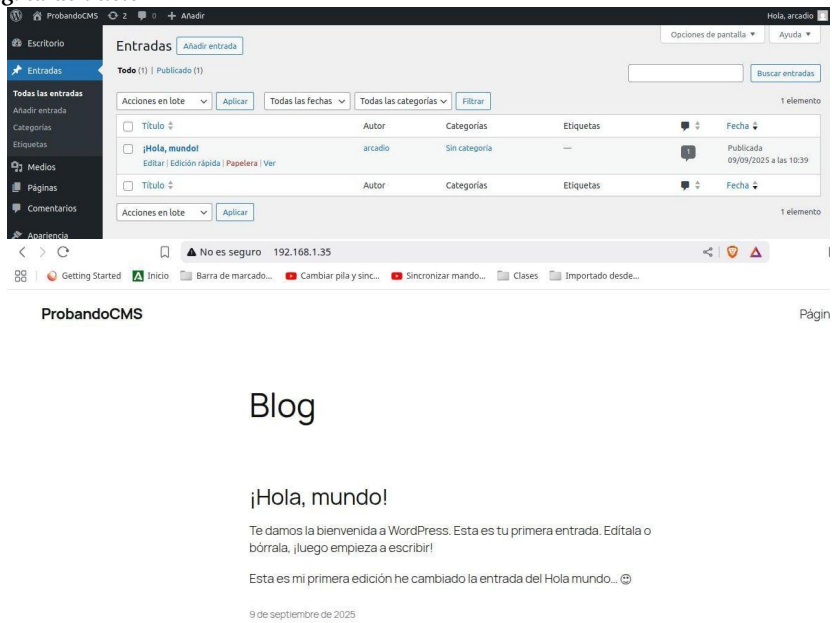


Tabla de contenidos

1. NUBE PRIVADA INTEGRAL
    1. Virtualización, Identidad Centralizada y Servicios de Próxima Generación
        1. Resumen
        2. Análisis del documento
2. AVISO LEGAL, EXENCIÓN DE RESPONSABILIDAD Y LICENCIA
    1. Carácter Educativo y Autoría
    2. Licencia de Uso
3. Server01
    1. Configurar KVM
    2. Utilidades varias
    3. Importante permisos
    4. Creamos MV y templates
    5. Utilidades
    6. Backups
4. Server01. Máquinas Virtuales
    1. MVs. Ldap. Authentik. Nextcloud
        1. Máquinas virtuales
        2. Certificados
        3. ldap + authentik
        4. Usuarios POSIX y Proxy authentik
        5. Authentik con ldap
        6. Configuración incial Máquinas virtuales
        7. Nextcloud
        8. Integración Authentik + Nextcloud con OIDC
        9. Crear nuevos usuarios ldap
        10. Automatizar Start/Stop máquinas virtuales
        11. dnsmasq.conf

    2. MVs. Bases de datos
        1. databases
    3. MVs. observability
        1. Grafana Loki
        2. n8n
        3. Clientes que suministran datos
5. Server01. OpenProject
    1. OpenProject
    2. Ldap y Openproject
        1. Usuarios posix
    3. n8n. MV: observability
        1. n8n
        2. Seguridad. Importante
    4. Trabajar con n8n
        1. Nodos (o conectores) esenciales de N8N, organizados en categorías lógicas
6. wordpress
    1. Instalación

Landmarks

1. Cover